

NoSQL Activity

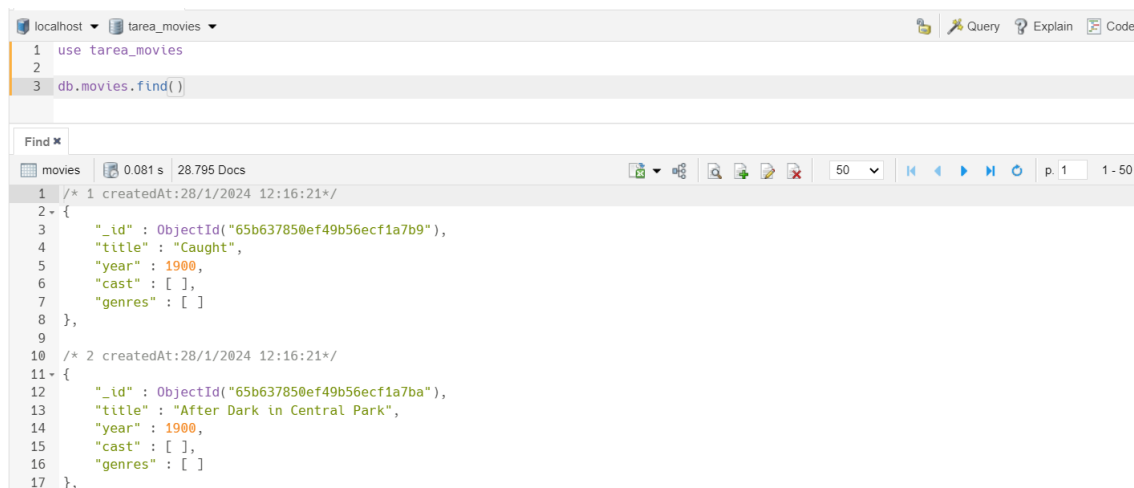
0. Import the 'movies' collection.

EXERCISE 1. Analysis using 'find'.

Query:

```
use tarea_movies  
db.movies.find()
```

Results:



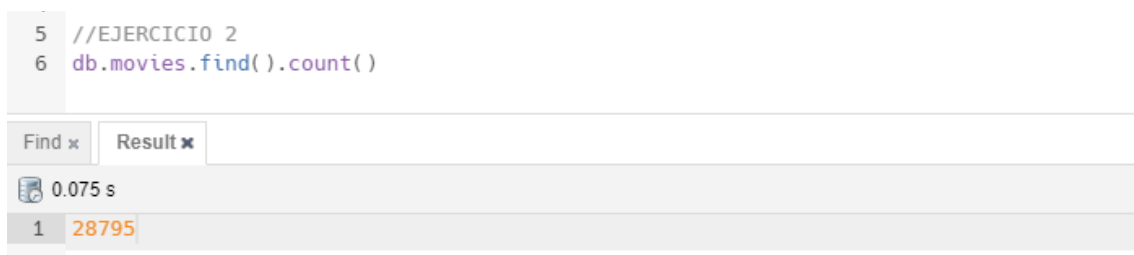
movies	0.081 s	28.795 Docs
1	/* 1 createdAt:28/1/2024 12:16:21*/	
2	{	
3	"_id" : ObjectId("65b637850ef49b56ecf1a7b9"),	
4	"title" : "Caught",	
5	"year" : 1900,	
6	"cast" : [],	
7	"genres" : []	
8	},	
9		
10	/* 2 createdAt:28/1/2024 12:16:21*/	
11	{	
12	"_id" : ObjectId("65b637850ef49b56ecf1a7ba"),	
13	"title" : "After Dark in Central Park",	
14	"year" : 1900,	
15	"cast" : [],	
16	"genres" : []	
17	},	

EXERCISE 2. Count documents in the collection.

Query:

```
db.movies.find().count()
```

Results:



5	//EJERCICIO 2
6	db.movies.find().count()

Find x	Result x
1	28795

EXERCISE 3. Insert a new document.

Query:

```
var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}  
db.movies.insertOne(newmovie)  
db.movies.find(newmovie)
```

Results:

```

8 //EJERCICIO 3
9 var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
10 db.movies.insertOne(newmovie)

```

Find x	Result x	Result (1) x
0.046 s		
Key	Value	Type
acknowledged	true	Bool
insertedid	65b66f07db5c4701d374f7a3	ObjectId

```

8 //EJERCICIO 3
9 var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
10 db.movies.insertOne(newmovie)
11 db.movies.find(newmovie)

```

Find x Result x Result (1) x Find (1) x

movies

0.156 s

1 Doc

EXERCISE 4. Delete the new document.**Query:**

```

var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
db.movies.deleteOne(newmovie)

```

```

var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
db.movies.find(newmovie)

```

Results:

```

13 //EJERCICIO 4
14 var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
15 db.movies.deleteOne(newmovie)

```

Find x	Result x	Result (1) x	Result (2) x
0.070 s			
Key	Value	Type	
acknowledged	true	Bool	
deletedCount	1	Int32	

```

16 var newmovie = {"title":"Remember the Titans", "year":2000,"cast":[ ], "genres":"Drama"}
17 db.movies.find(newmovie)

```

Find x	Result x	Result (1) x	Result (2) x	Error x	Find (1) x
movies 0.065 s 0 Doc					
Key	Value				

No records found

EXERCISE 5. Count how many movies have actors named "and".**Query:**

```

var query1 = {"cast":"and"}
var fase1 = {$match:query1}
var query2 = {"_id":"and", "numero":{"$sum:1}}
var fase2 = {$group:query2}
var etapas = [fase1, fase2]
db.movies.aggregate(etapas)

```

Results:

```

20 //EJERCICIO 5
21 var query1 = {"cast":"and"}
22 var fase1 = {$match:query1}
23 var query2 = {"_id":"and", "numero":{"$sum:1}}
24 var fase2 = {$group:query2}
25 var etapas = [fase1, fase2]
26 db.movies.aggregate(etapas)
27

```

Result x Find (1) x Aggregate (4) x

movies 0.121 s 1 Doc

	_id	numero
1	and	93

EXERCISE 6. Extract the value "and" from the array without deleting the document.

Query:

```

var query3 = { }
var operacion = {$pull:{"cast":"and"}}
db.movies.updateMany(query3, operacion)

```

Results:

```

29 //EJERCICIO 6
30 var query3 = { }
31 var operacion = {$pull:{"cast":"and"}}
32 db.movies.updateMany(query3, operacion)
33

```

Aggregate (1) x Result x

0.305 s

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 28795, modifiedCount : 93 }	Object
acknowledged	true	Bool
matchedCount	28.795 (28.8K)	Int32
modifiedCount	93	Int32

EXERCISE 7. Count documents where the "cast" array is empty.

Query:

```

var query4 = {"cast":[]}
var fase1 = {$match:query4}
var fase2 = {$count:"total_vacio"}
var etapas = [fase1, fase2]
db.movies.aggregate(etapas)

```

Results:

```

34 //EJERCICIO 7
35 var query4 = {"cast":[]}
36 var fase1 = {$match:query4}
37 var fase2 = {$count:"total_vacio"}
38 var etapas = [fase1, fase2]
39 db.movies.aggregate(etapas)
40

```

Aggregate (1) x	Result x	Find x	Aggregate (2) x
movies	0.103 s	1 Doc	
total_vacio			
1	986		

EXERCISE 8. Add "Undefined" to empty "cast" arrays.

Query:

```

var query5 = {"cast":[]}
var operacion = {$set:{"cast":["Undefined"]}}
db.movies.updateMany(query5, operacion)

```

Results:

```

41 //EJERCICIO 8
42 var query5 = {"cast":[]}
43 var operacion = {$set:{"cast":["Undefined"]}}
44 db.movies.updateMany(query5, operacion)
45

```

Aggregate (1) x	Result x	Find x	Aggregate (2) x	Result (1) x
0.175 s				
Key	Value	Type		
(1)	{ acknowledged : true, matchedCount : 986, modifiedCount : 986 }	Object		
acknowledged	true	Bool		
matchedCount	986	Int32		
modifiedCount	986	Int32		

```

45
46 db.movies.find()
47

```

Aggregate (1) x	Result x	Find x	Aggregate (2) x	Result (1) x	Find (1) x
movies	0.029 s	28.795 Docs			
1	/* 1 createdAt:28/1/2024 12:16:21*/				
2	{				
3	"_id" : ObjectId("65b637850ef49b56ecf1a7b9"),				
4	"title" : "Caught",				
5	"year" : 1900,				
6	"cast" : ["Undefined"],				
7	"genres" : []				
8	},				
9					
10	/* 2 createdAt:28/1/2024 12:16:21*/				
11	{				
12	"_id" : ObjectId("65b637850ef49b56ecf1a7ba"),				
13	"title" : "After Dark in Central Park",				
14	"year" : 1900,				
15	"cast" : ["Undefined"],				
16	"genres" : []				
17	},				
18					

EXERCISE 9. Count how many documents have an empty "genres" array.

Query:

```
var query6 = {"genres":[]}
var fase1 = {$match:query6}
var fase2 = {$count:"total_vacio"}
var etapas = [fase1, fase2]
db.movies.aggregate(etapas)
```

Results:

```
46 //EJERCICIO 9
47 var query6 = {"genres":[]}
48 var fase1 = {$match:query6}
49 var fase2 = {$count:"total_vacio"}
50 var etapas = [fase1, fase2]
51 db.movies.aggregate(etapas)
```

Find x Aggregate x	
movies	0.084 s 1 Doc
total_vacio	
1	901

EXERCISE 10. Add "Undefined" to empty "genres" arrays.

Query:

```
var query7 = {"genres":[]}
var operacion = {$set:{"genres":["Undefined"]}}
db.movies.updateMany(query7, operacion)
```

Results:

```
53 //EJERCICIO 10
54 var query7 = {"genres":[]}
55 var operacion = {$set:{"genres":["Undefined"]}}
56 db.movies.updateMany(query7, operacion)
```

Find x Aggregate x Result x		
0.140 s		
Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 901, modifiedCount: 901 }	Object
acknowledged	true	Bool
matchedCount	901	Int32
modifiedCount	901	Int32

```
58 db.movies.find()
```

Find x Aggregate x Result x Find (1) x			
movies	0.029 s	28.795 Docs	
836	},		
837			
838	/* 94 createdAt:28/1/2024 12:16:21*/		
839	{		
840	"_id" : ObjectId("65b637850ef49b56ecf1a816"),		
841	"title" : "Wonderful Trick Donkey, The",		
842	"year" : 1901,		
843	"cast" : ["Undefined"],		
844	"genres" : ["Undefined"]		
845	},		
846			
847	/* 95 createdAt:28/1/2024 12:16:21*/		
848	{		
849	"_id" : ObjectId("65b637850ef49b56ecf1a817"),		
850	"title" : "Arrival of Prince Henry (of Prussia) and President Roosevelt at Shooter's Island (1902)",		
851	"year" : 1902,		
852	"cast" : ["Undefined"],		
853	"genres" : ["Short"]		
854	},		
855			

EXERCISE 11. Most recent year in the collection.**Query:**

```
var query8 = {}
var proyeccion = {"year": 1, "_id":0}
db.movies.find(query8,proyeccion).sort({"year":-1}).limit(1)
```

Results:

```
58 //EJERCICIO 11
59 var query8 = {}
60 var proyeccion = {"year": 1, "_id":0}
61 db.movies.find(query8,proyeccion).sort({"year":-1}).limit(1)
62
```

Find x	Aggregate x	Result x	Find (1) x	Find (2) x
movies 0.066 s 1 Doc				
1	{			
2	"year" : 2018			
3	}			

EXERCISE 12. Number of movies from the last 20 years.**Query:**

```
var query9 = {"_id":"$year", "movies":{"$sum:1}}
var fase1 = {$group:query9}
var fase2 ={$sort:{"_id":-1}}
var fase3 ={$limit:20}
var query10 = {"_id":null, "total":{"$sum:"$movies"}}
var fase4 = {$group:query10}
var etapas = [fase1, fase2,fase3,fase4]
db.movies.aggregate(etapas)
```

Results:

```
63 //EJERCICIO 12
64 var query9 = {"_id":"$year", "movies":{"$sum:1}}
65 var fase1 = {$group:query9}
66 var fase2 ={$sort:{"_id":-1}}
67 var fase3 ={$limit:20}
68 var query10 = {"_id":null, "total":{"$sum:"$movies"}}
69 var fase4 = {$group:query10}
70 var etapas = [fase1, fase2,fase3,fase4]
71 db.movies.aggregate(etapas)
72
```

Find x	Aggregate x	Result x	Find (1) x	Find (2) x
movies 0.088 s 1 Doc				
1	{			
2	"_id" : null,			
3	"total" : 4787			
4	}			

EXERCISE 13. Number of movies from the 1960s.**Query:**

```

var query11 = {"year":{"$gte:1960, $lte:1969}}
var fase1 = {$match:query11}
var query12 = {"_id":null, "total":{"$sum:1}}
var fase2 = {$group:query12}
var etapas = [fase1,fase2]
db.movies.aggregate(etapas)

```

Results:

```

73 //EJERCICIO 13
74 var query11 = {"year":{"$gte:1960, $lte:1969}}
75 var fase1 = {$match:query11}
76 var query12 = {"_id":null, "total":{"$sum:1}}
77 var fase2 = {$group:query12}
78 var etapas = [fase1,fase2]
79 db.movies.aggregate(etapas)
80

```

EXERCISE 14. Year or years with the most movies.

First, we execute phase 1 and phase 2. After reviewing the results, we set the "limit". In this case, there are no repetitions.

Query:

```

var query13 = {"_id":"$year", "movies":{"$sum:1}}
var fase1 = {$group:query13}
var fase2 = {$sort:{"movies":-1}}
var fase3 = {$limit:1}
var etapas = [fase1,fase2,fase3]
db.movies.aggregate(etapas)

```

Results:

```

--
81 //EJERCICIO 14
82 var query13 = {"_id":"$year", "movies":{"$sum:1}}
83 var fase1 = {$group:query13}
84 var fase2 = {$sort:{"movies":-1}}
85 var fase3 = {$limit:1}
86 var etapas = [fase1,fase2,fase3]
87 db.movies.aggregate(etapas)
88

```

EXERCISE 15. Year or years with the fewest movies.

First, we execute phase 1 and phase 2. After reviewing the results, we set the "limit". In this case, three years are repeated, so we set limit 3.

Query:

```
var query14 = {"_id": "$year", "movies": {$sum:1}}
var fase1 = {$group:query14}
var fase2 = {$sort:{"movies":1}}
var fase3 = {$limit:3}
var etapas = [fase1,fase2,fase3]
db.movies.aggregate(etapas)
```

Results:

```
89 //EJERCICIO 15
90 var query14 = {"_id": "$year", "movies": {$sum:1}}
91 var fase1 = {$group:query14}
92 var fase2 = {$sort:{"movies":1}}
93 var fase3 = {$limit:3}
94 var etapas = [fase1,fase2,fase3]
95 db.movies.aggregate(etapas)
```

Aggregate x	Aggregate (5) x	Aggregate (1) x	Aggregate (2) x
movies	0.063 s	3 Docs	
1 /* 1 */			
2 {			
3 "_id" : 1907,			
4 "movies" : 7			
5 },			
6			
7 /* 2 */			
8 {			
9 "_id" : 1906,			
10 "movies" : 7			
11 },			
12			
13 /* 3 */			
14 {			
15 "_id" : 1902,			
16 "movies" : 7			
17 }			

EXERCISE 16. New collection "actors".**Query:**

```
var fase1 = {$unwind:"$cast"}
var query15 = {"_id":0}
var fase2 = {$project:query15}
var fase3 = {$out: "actors"}
var etapas = [fase1, fase2, fase3]
db.movies.aggregate(etapas)
```

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


Results:

```

97 //EJERCICIO 16
98 var fase1 = {$unwind:"$cast"}
99 var query15 = {"_id":0}
100 var fase2 = {$project:query15}
101 var fase3 = {$out: "actors"}
102 var etapas = [fase1, fase2, fase3]
103 db.movies.aggregate(etapas)
104
105 db.actors.find().count()

```

Aggregate x	Aggregate (5) x	Aggregate (1) x	Aggregate (2) x	Result x	Find x	Result (1) x
0.028 s						
1	83224					

EXERCISE 17: Based on the actors collection (new collection), display a list of the top 5 actors who have participated in the most movies, showing the number of movies they have appeared in.

Query:

```

var query16 = {"cast":{"$ne:"Undefined"}}
var fase1 = {$match:query16}
var query17 = {"_id":"$cast", "totalmovies":{"$sum:1}}
var fase2 = {$group:query17}
var fase3 = {$sort:{"totalmovies":-1}}
var fase4 = {$limit:5}
var etapas =[fase1, fase2, fase3, fase4]
db.actors.aggregate(etapas)

```

Results:

```

107 ///EJERCICIO 17
108 var query16 = {"cast":{"$ne:"Undefined"}}
109 var fase1 = {$match:query16}
110 var query17 = {"_id":"$cast", "totalmovies":{"$sum:1}}
111 var fase2 = {$group:query17}
112 var fase3 = {$sort:{"totalmovies":-1}}
113 var fase4 = {$limit:5}
114 var etapas =[fase1, fase2, fase3, fase4]
115 db.actors.aggregate(etapas)

```

Find x	Aggregate x
actors 0.532 s 5 Docs	
_id	totalmovies
1 Harold Lloyd	190
2 Hoot Gibson	142
3 John Wayne	136
4 Charles Starrett	116
5 Bebe Daniels	103

```

107 ///EJERCICIO 17
108 var query16 = {"cast":{"$ne":"Undefined"}}
109 var fase1 = {$match:query16}
110 var query17 = {"_id":"$cast", "totalmovies":{"$sum:1}}
111 var fase2 = {$group:query17}
112 var fase3 = {$sort:{"totalmovies":-1}}
113 var fase4 = {$limit:5}
114 var etapas =[fase1, fase2, fase3, fase4]
115 db.actors.aggregate(etapas)
116

```

Find x	Aggregate x
actors	0.532 s 5 Docs
1	/* 1 */
2	{
3	"_id" : "Harold Lloyd",
4	"totalmovies" : 190
5	},
6	
7	/* 2 */
8	{
9	"_id" : "Hoot Gibson",
10	"totalmovies" : 142
11	},
12	
13	/* 3 */
14	{
15	"_id" : "John Wayne",
16	"totalmovies" : 136
17	},
18	
19	/* 4 */
20	{
21	"_id" : "Charles Starrett",
22	"totalmovies" : 116
23	},
24	
25	/* 5 */
26	{
27	"_id" : "Bebe Daniels",
28	"totalmovies" : 103
29	}

EXERCISE 18: Based on the actors collection (new collection), group by movie and year, displaying the top 5 movies with the highest number of actors, showing the total number of actors in each.

Query:

```

var query18 = {"_id":{"title":"$title", "year":"$year"},"cuenta":{"$addToSet":"$cast"}}
var fase1 = {$group:query18}
var query19 = {"_id":{"title":"$_id.title", "year":"$_id.year"}, "cuenta":{"$size":"$cuenta"}}
var fase2= {$project:query19}
var fase3 = {$sort:{"cuenta":-1}}
var fase4 ={$limit:5}
var etapas = [fase1, fase2, fase3, fase4]
db.actors.aggregate(etapas)

```

Results:

```

117 //EJERCICIO 18
118 var query18 = {"_id":{"title":"$title", "year":"$year"},"cuenta":{"addToSet":"$cast"}}
119 var fase1 = {$group:query18}
120 var query19 = {"_id":{"title":"$_id.title", "year":"$_id.year"}, "cuenta":{"$size: "$cuenta"}}
121 var fase2= {$project:query19}
122 var fase3 = {$sort:{"cuenta":-1}}
123 var fase4 = {$limit:5}
124 var etapas = [fase1, fase2, fase3, fase4]
125 db.actors.aggregate(etapas)

```

		_id	cuenta
	title	year	
1	The Twilight Saga: Breaking Dawn - Part 2	2012	35
2	Anchorman 2: The Legend Continues	2013	33
3	Cars 2	2011	32
4	Avengers: Infinity War	2018	29
5	Grown Ups 2	2013	28

	actors	0.602 s	5 Docs
1	/* 1 */		
2	{		
3	"_id" : {		
4	"title" : "The Twilight Saga: Breaking Dawn - Part 2",		
5	"year" : 2012		
6	},		
7	"cuenta" : 35		
8	},		
9			
10	/* 2 */		
11	{		
12	"_id" : {		
13	"title" : "Anchorman 2: The Legend Continues",		
14	"year" : 2013		
15	},		
16	"cuenta" : 33		
17	},		
18			
19	/* 3 */		
20	{		
21	"_id" : {		
22	"title" : "Cars 2",		
23	"year" : 2011		
24	},		
25	"cuenta" : 32		
26	},		
27			
28	/* 4 */		
29	{		
30	"_id" : {		
31	"title" : "Avengers: Infinity War",		
32	"year" : 2018		
33	},		
34	"cuenta" : 29		
35	},		
36			
37	/* 5 */		
38	{		
39	"_id" : {		
40	"title" : "Grown Ups 2",		
41	"year" : 2013		
42	},		
43	"cuenta" : 28		
44	}		

EXERCISE 19: Based on the actors collection (new collection), display the top 5 actors with the longest careers. The output should include the year their career started, the year it ended, and the total number of years they worked.

Query:

```
var query20 = {"cast":{"$ne:"Undefined"}}
var fase1 = {$match:query20}
var query21 = {"_id":{"$cast"}, "comienza": {$min:"$year"}, "termina":{"$max:"$year"}}
var fase2 = {$group:query21}
var query22= {"años":{"$subtract":["$termina", "$comienza"]}}
var fase3= {$addFields:query22}
var fase4= {$sort:{ "años":-1}}
var fase5= {$limit:5}
var etapas= [fase1, fase2, fase3, fase4, fase5]
db.actors.aggregate(etapas)
```

Results:

```
127 //EJERCICIO 19
128 var query20 = {"cast":{"$ne:"Undefined"}}
129 var fase1 = {$match:query20}
130 var query21 = {"_id":{"$cast"}, "comienza": {$min:"$year"}, "termina":{"$max:"$year"}}
131 var fase2 = {$group:query21}
132 var query22= {"años":{"$subtract":["$termina", "$comienza"]}}
133 var fase3= {$addFields:query22}
134 var fase4= {$sort:{ "años":-1}}
135 var fase5= {$limit:5}
136 var etapas= [fase1, fase2, fase3, fase4, fase5]
137 db.actors.aggregate(etapas)
138
```

Aggregate x		Aggregate (1) x	
actors	0.412 s	5 Docs	
_id	comienza	termina	años
1 Harrison Ford	1919 (1.9K)	2017 (2.0K)	98
2 Gloria Stuart	1932 (1.9K)	2012 (2.0K)	80
3 Kenny Baker	1937 (1.9K)	2012 (2.0K)	75
4 Lillian Gish	1912 (1.9K)	1987 (2.0K)	75
5 Mickey Rooney	1932 (1.9K)	2006 (2.0K)	74

Aggregate x		Aggregate (1) x	
 actors	 0.412 s	5 Docs	

```
1  /* 1 */
2  {
3      "_id" : "Harrison Ford",
4      "comienza" : 1919,
5      "termina" : 2017,
6      "años" : 98
7  },
8
9  /* 2 */
10 {
11     "_id" : "Gloria Stuart",
12     "comienza" : 1932,
13     "termina" : 2012,
14     "años" : 80
15 },
16
17 /* 3 */
18 {
19     "_id" : "Kenny Baker",
20     "comienza" : 1937,
21     "termina" : 2012,
22     "años" : 75
23 },
24
25 /* 4 */
26 {
27     "_id" : "Lillian Gish",
28     "comienza" : 1912,
29     "termina" : 1987,
30     "años" : 75
31 },
32
33 /* 5 */
34 {
35     "_id" : "Mickey Rooney",
36     "comienza" : 1932,
37     "termina" : 2006,
38     "años" : 74
39 }
```

EXERCISE 20: Based on the actors collection (new collection), save the data into a new collection called "genres" by performing the '\$unwind' stage on the "genres" field. Then, count the number of documents in the new collection.

Query:

```
var fase1 = {$unwind:"$genres"}
var query23 = {"_id":0}
var fase2 = {$project:query23}
var fase3 = {$out: "genres"}
var etapas = [fase1, fase2, fase3]
db.actors.aggregate(etapas)
```

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

Results:

```

139 //EJERCICIO 20
140 var fase1 = {$unwind:"$genres"}
141 var query23 = {"_id":0}
142 var fase2 = {$project:query23}
143 var fase3 = {$out: "genres"}
144 var etapas = [fase1, fase2, fase3]
145 db.actors.aggregate(etapas)
146
147 db.genres.find().count()

```

Aggregate x	Aggregate (1) x	Aggregate (2) x	Result x
0.028 s			
1	104950		

EXERCISE 21: Based on the genres collection (new collection), display the top 5 records grouped by "Year and Genre" that have the highest number of different movies, showing the total number of movies in each group.

Query:

```

var query24 = {"_id":{"year":"$year", "genres":"$genres"},"movies":{"addToSet":"$title"}}
var fase1 = {$group:query24}
var query25 = {"_id":{"year":"$_id.year", "genres":"$_id.genres"}, "movies":{"size":
"$movies"}}
var fase2= {$project:query25}
var fase3 = {$sort:{"movies":-1}}
var fase4 ={$limit:5}
var etapas = [fase1, fase2, fase3, fase4]
db.genres.aggregate(etapas)

```

Results:

```

149 //EJERCICIO 21
150 var query24 = {"_id":{"year":"$year", "genres":"$genres"},"movies":{"addToSet":"$title"}}
151 var fase1 = {$group:query24}
152 var query25 = {"_id":{"year":"$_id.year", "genres":"$_id.genres"}, "movies":{"size": "$movies"}}
153 var fase2= {$project:query25}
154 var fase3 = {$sort:{"movies":-1}}
155 var fase4 ={$limit:5}
156 var etapas = [fase1, fase2, fase3, fase4]
157 db.genres.aggregate(etapas)

```

Aggregate x	Aggregate (1) x	Aggregate (2) x	Result x	Aggregate (3) x	Aggregate (4) x
genres 0.254 s 5 Docs					
		_id			
	year	genres	movies		
1	1919	Drama	291		
2	1925	Drama	247		
3	1924	Drama	233		
4	1919	Comedy	226		
5	1922	Drama	209		

```

genres 0.254 s 5 Docs
1 /* 1 */
2 {
3   "_id" : {
4     "year" : 1919,
5     "genres" : "Drama"
6   },
7   "movies" : 291
8 },
9
10 /* 2 */
11 {
12   "_id" : {
13     "year" : 1925,
14     "genres" : "Drama"
15   },
16   "movies" : 247
17 },
18
19 /* 3 */
20 {
21   "_id" : {
22     "year" : 1924,
23     "genres" : "Drama"
24   },
25   "movies" : 233
26 },
27
28 /* 4 */
29 {
30   "_id" : {
31     "year" : 1919,
32     "genres" : "Comedy"
33   },
34   "movies" : 226
35 },
36
37 /* 5 */
38 {
39   "_id" : {
40     "year" : 1922,
41     "genres" : "Drama"
42   },
43   "movies" : 209
44 }

```

EXERCISE 22: Using the genres collection (new collection), display the top 5 actors who have participated in the most different genres, along with the genres they have acted in. The total number of different genres each actor has participated in should be shown. No data is deleted from the collection, only filtered to exclude certain records.

Query:

```

var query24 = {"cast": {$ne:"Undefined"}, "genres":{$ne:"Undefined"}}
var fase1 = {$match:query24}
var query25 = {"_id":"$cast", "genres":{$addToSet:"$genres"}}
var fase2 = {$group:query25}
var query26 = {"_id":"$_id", "numgeneros":{$size:"$genres"}, "genres":"$genres"}
var fase3 = {$project:query26}
var fase4 = {$sort: {"numgeneros":-1}}
var fase5 = {$limit:5}
var etapas = [fase1, fase2, fase3, fase4, fase5]
db.genres.aggregate(etapas)

```

Results:

```
159 //EJERCICIO 22
160 var query24 = {"cast": {$ne:"Undefined"}, "genres":{$ne:"Undefined"}}
161 var fase1 = {$match:query24}
162 var query25 = {"_id": "$cast", "genres":{$addToSet:"$genres"}}
163 var fase2 = {$group:query25}
164 var query26 = {"_id": "$_id", "numgeneros":{$size:"$genres"}, "genres": "$genres"}
165 var fase3 = {$project:query26}
166 var fase4 = {$sort: {"numgeneros":-1}}
167 var fase5 = {$limit:5}
168 var etapas = [fase1, fase2, fase3, fase4, fase5]
169 db.genres.aggregate(etapas)
```

Aggregate x Aggregate (1) x Aggregate (2) x Result x Aggregate (3) x Aggregate (4) x Aggregate (5) x

genres 0.454 s 5 Docs

	_id	numgeneros	genres
1	Dennis Quaid	20	Array[20]
2	Colin Farrell	18	Array[18]
3	Helen Mirren	18	Array[18]
4	Michael Peña	18	Array[18]
5	James Mason	18	Array[18]

genres	0.454 s	5 Docs
1 /* 1 */		
2 {		
3 "_id" : "Dennis Quaid",		
4 "numgeneros" : 20,		
5 "genres" : [
6 "Fantasy",		
7 "Family",		
8 "Dance",		
9 "Satire",		
10 "Adventure",		
11 "Horror",		
12 "Thriller",		
13 "Drama",		
14 "Animated",		
15 "Crime",		
16 "Biography",		
17 "Disaster",		
18 "Romance",		
19 "Action",		
20 "Musical",		
21 "Sports",		
22 "Comedy",		
23 "Science Fiction",		
24 "Western",		
25 "Suspense"		
26]		
27 },		

genres	0.454 s	5 Docs
29 /* 2 */		
30 {		
31 "_id" : "Colin Farrell",		
32 "numgeneros" : 18,		
33 "genres" : [
34 "Musical",		
35 "Horror",		
36 "Adventure",		
37 "Mystery",		
38 "Superhero",		
39 "Historical",		
40 "Crime",		
41 "Animated",		
42 "Thriller",		
43 "Drama",		
44 "Action",		
45 "Noir",		
46 "Fantasy",		
47 "Family",		
48 "Science Fiction",		
49 "War",		
50 "Comedy",		
51 "Western"		
52]		
53 },		


```

genres 0.454 s 5 Docs
55 /* 3 */
56 {
57   "_id" : "Helen Mirren",
58   "numgeneros" : 18,
59   "genres" : [
60     "Horror",
61     "Adventure",
62     "Erotic",
63     "Historical",
64     "Crime",
65     "Political",
66     "Animated",
67     "Drama",
68     "Thriller",
69     "Action",
70     "Romance",
71     "Spy",
72     "Biography",
73     "Fantasy",
74     "Family",
75     "Science Fiction",
76     "Comedy",
77     "Mystery"
78   ]
79 },

```

```

genres 0.454 s 5 Docs
80
81 /* 4 */
82 {
83   "_id" : "Michael Peña",
84   "numgeneros" : 18,
85   "genres" : [
86     "Musical",
87     "Horror",
88     "Adventure",
89     "Superhero",
90     "Crime",
91     "Animated",
92     "Thriller",
93     "Action",
94     "Drama",
95     "Martial Arts",
96     "Biography",
97     "Fantasy",
98     "Suspense",
99     "Family",
100    "Science Fiction",
101    "War",
102    "Comedy",
103    "Mystery"
104  ]
105 },

```

```

genres 0.454 s 5 Docs
105 },
106
107 /* 5 */
108 {
109   "_id" : "James Mason",
110   "numgeneros" : 18,
111   "genres" : [
112     "Suspense",
113     "Short",
114     "Fantasy",
115     "Adventure",
116     "Thriller",
117     "Animated",
118     "Drama",
119     "Crime",
120     "Biography",
121     "Romance",
122     "Noir",
123     "Action",
124     "Musical",
125     "Western",
126     "War",
127     "Science Fiction",
128     "Comedy",
129     "Mystery"
130   ]
131 }

```

EXERCISE 23: Using the genres collection (new collection), display the top 5 movies and their corresponding year that have been categorized under the most different genres. The genres and the total number of genres for each movie should also be shown.

Query:

```
var query27 = {"_id":{"title":"$title", "year":"$year"}, "genres":{"addToSet":"$genres"}}
var fase1 = {$group:query27}
var query28 = {"_id":1, "title":"$id.title", "numgenres":{"size":"$genres"},
"genres":"$genres"}
var fase2 = {$project:query28}
var fase3 = {$sort:{"numgenres":-1}}
var fase4 = {$limit: 5}
var etapas = [fase1, fase2, fase3, fase4]
db.genres.aggregate(etapas)
```

Results:

```
171 //EJERCICIO 23
172 var query27 = {"_id":{"title":"$title", "year":"$year"}, "genres":{"addToSet":"$genres"}}
173 var fase1 = {$group:query27}
174 var query28 = {"_id":1, "title":"$id.title", "numgenres":{"size":"$genres"}, "genres":"$genres"}
175 var fase2 = {$project:query28}
176 var fase3 = {$sort:{"numgenres":-1}}
177 var fase4 = {$limit: 5}
178 var etapas = [fase1, fase2, fase3, fase4]
179 db.genres.aggregate(etapas)
```

	title	year	numgenres	genres
1	American Made	2017	7	Array[7]
2	Dunkirk	2017	6	Array[6]
3	Wonder Woman	2017	6	Array[6]
4	Thor: Ragnarok	2017	6	Array[6]
5	My Little Pony: The Movie	2017	6	Array[6]

	numgenres	genres
1	7	["Biography", "Drama", "Crime", "Comedy", "Historical", "Thriller", "Action"]
2	6	["War", "Historical", "Action", "Adventure", "Drama", "Thriller"]

genres	0.532 s	5 Docs
36 /* 3 */		
37 {		
38 "_id" : {		
39 "title" : "Wonder Woman",		
40 "year" : 2017		
41 },		
42 "numgenres" : 6,		
43 "genres" : [
44 "Superhero",		
45 "War",		
46 "Drama",		
47 "Adventure",		
48 "Action",		
49 "Fantasy"		
50]		
51 },		
52		
53 /* 4 */		
54 {		
55 "_id" : {		
56 "title" : "Thor: Ragnarok",		
57 "year" : 2017		
58 },		
59 "numgenres" : 6,		
60 "genres" : [
61 "Fantasy",		
62 "Action",		
63 "Adventure",		
64 "Science Fiction",		
65 "Comedy",		
66 "Superhero"		
67]		
68 },		

genres	0.532 s	5 Docs
69		
70 /* 5 */		
71 {		
72 "_id" : {		
73 "title" : "My Little Pony: The Movie",		
74 "year" : 2017		
75 },		
76 "numgenres" : 6,		
77 "genres" : [
78 "Comedy",		
79 "Animated",		
80 "Musical",		
81 "Fantasy",		
82 "Family",		
83 "Adventure"		
84]		
85 }		

EXERCISE 24. Calculate the average number of movies made per year in the 21st century (up to the most recent date in the collection).

Query:

```

var query1 = {"year": {$gte:2000, $lte:2018}}
var fase1 = {$match:query1}
var query2 = {"_id":{"$year", "movies":{$sum:1}}}
var fase2 = {$group:query2}
var query3 = {"_id": null, "media_movies":{$avg:"$movies"}}
var fase3 = {$group:query3}
var etapas = [fase1, fase2, fase3]
db.movies.aggregate(etapas)

```

Results:

```

181 //EJERCICIO 24
182 var query1 = {"year": {$gte:2000, $lte:2018}}
183 var fase1 = {$match:query1}
184 var query2 = {"_id":"$year", "movies":{$sum:1}}
185 var fase2 = {$group:query2}
186 var query3 = {"_id": null, "media_movies":{$avg:"$movies"}}
187 var fase3 = {$group:query3}
188 var etapas = [fase1, fase2, fase3]
189 db.movies.aggregate(etapas)

```

Aggregate x	Aggregate (1) x	Aggregate (2) x	Aggregate (7) x
movies	0.080 s	1 Doc	
1	{		
2	"_id" : null,		
3	"media_movies" : 240.31578947368422		
4	}		

EXERCISE 25: Query which actor has made the most movies in each movie genre, showing the number of movies they have made.

Query:

```

var query1 = {"cast":{"$ne:"Undefined"}, "genres":{"$ne:"Undefined"}}
var fase1 = {$match:query1}
var query2 = {"_id":{"genre":"$genres", "actor":"$cast"}, "movies":{$sum:1}}
var fase2 = {$group:query2}
var fase3 = {$sort:{"_id.genre":1, "movies":-1}}
var query3 = {"_id":"$_id.genre", "topactor":{"$first:"$_id.actor"},
"movies":{"$first:"$movies"}}
var fase4 = {$group:query3}
var etapas = [fase1,fase2,fase3,fase4]
db.genres.aggregate(etapas)

```

Results:

```

192 //EJERCICIO 25
193 var query1 = {"cast":{"$ne":"Undefined"}, "genres":{"$ne":"Undefined"}}
194 var fase1 = {$match:query1}
195 var query2 = {"_id":{"genre":"$genres", "actor":"$cast"}, "movies":{"$sum:1}}
196 var fase2 = {$group:query2}
197 var fase3 = {$sort:{"_id.genre":1, "movies":-1}}
198 var query3 = {"_id":"$_id.genre", "topactor":{"$first:$_id.actor"}, "movies":{"$first:$movies"}}
199 var fase4 = {$group:query3}
200 var etapas = [fase1,fase2,fase3,fase4]
201 db.genres.aggregate(etapas)
202

```

Aggregate x

Aggregate (8) x

Aggregate (9) x

genres	0.756 s	40 Docs
_id	topactor	movies
1 Satire	Anna Faris	3
2 Adventure	Johnny Weissmuller	29
3 Comedy	Harold Lloyd	186
4 Political	George Clooney	3
5 Legal	Chris Evans	1
6 Noir	Edward G. Robinson	14
7 War	John Wayne	12
8 Independent	Sarah Prikyri	1
9 Silent	Conway Tearle	1
10 Documentary	Iraq War	5
11 Historical	Tyrone Power	5
12 Musical	Bing Crosby	41
13 Short	The Three Stooges	64
14 Disaster	George Kennedy	4
15 Sport	Peggy Moran	1
16 Teen	Aundrea Fares	2
17 Performance	Jonas Brothers	2
18 Spy	Dean Martin	5
19 Fantasy	Emma Watson	9
20 Supernatural	Vera Farmiga	2
21 Western	Hoot Gibson	131
22 Family	Maggie Smith	7
23 Horror	Boris Karloff	28
24 Science Fiction	William Shatner	9
25 Martial Arts	Jackie Chan	3
26 Slasher	Robert Englund	2
27 Dance	Adam G. Sevani	3
28 Crime	Edward G. Robinson	21
29 Biography	Ed Harris	8
30 Action	Jean-Claude Van Damme	22
31 Suspense	Cary Grant	4
32 Sports	Elyse Knox	4
33 Romance	Gary Cooper	15
34 Animated	Tom and Jerry	85
35 Superhero	Chris Evans	8
36 Thriller	Nicolas Cage	14
37 Drama	Bette Davis	56
38 Erotic	Jamie Gillis	4
39 Mystery	Warner Oland	16
40 Live Action	Neil Patrick Harris	2

EXERCISE 26: Calculate the total number of movies and actors who have participated in each movie genre, ordered in descending order by the number of movies.

Query:

```
var query1 = {"cast":{"$ne":"Undefined"}, "genres":{"$ne":"Undefined"}}
var fase1 = {$match:query1}
var query2 = {"_id":"$genres", "total_movies":{"$sum:1},
"total_actors":{"addToSet":"$cast"}}
var fase2 = {$group:query2}
var query3 = {"_id":0, "genre":"$ _id", "total_movies":"$total_movies",
"total_actors":{"$size":"$total_actors"}}
var fase3 = {$project:query3}
var fase4 = {$sort:{"total_movies":-1}}
var etapas = [fase1, fase2, fase3, fase4]
db.genres.aggregate(etapas)
```

Results:

```

204 //EJERCICIO 26
205 var query1 = {"cast":{"$ne":"Undefined"}, "genres":{"$ne":"Undefined"}}
206 var fase1 = {$match:query1}
207 var query2 = {"_id":"$genres", "total_movies":{"$sum:1}, "total_actors":{"addToSet":"$cast"}}
208 var fase2 = {$group:query2}
209 var query3 = {"_id":0, "genre":"$ _id", "total_movies":"$total_movies", "total_actors":{"$size":"$total_actors"}}
210 var fase3 = {$project:query3}
211 var fase4 = {$sort:{"total_movies":-1}}
212 var etapas = [fase1, fase2, fase3, fase4]
213 db.genres.aggregate(etapas)
214

```

Aggregate (8) x Aggregate (9) x Aggregate (1) x Aggregate (2) x			
genres 0.285 s 40 Docs			
	genre	total_movies	total_actors
1	Drama	26.174 (26.2K)	7489 (7.5K)
2	Comedy	23.276 (23.3K)	6906 (6.9K)
3	Western	7289 (7.3K)	2201 (2.2K)
4	Crime	4779 (4.8K)	2562 (2.6K)
5	Action	4671 (4.7K)	2670 (2.7K)
6	Horror	3853 (3.9K)	2911 (2.9K)
7	Romance	3803 (3.8K)	2183 (2.2K)
8	Thriller	3616 (3.6K)	2387 (2.4K)
9	Musical	3525 (3.5K)	1841 (1.8K)
10	Adventure	3309 (3.3K)	2075 (2.1K)
11	Science Fiction	2857 (2.9K)	1988 (2.0K)
12	Animated	1869 (1.9K)	1167 (1.2K)
13	Family	1808 (1.8K)	1285 (1.3K)
14	Mystery	1715 (1.7K)	1076 (1.1K)
15	Fantasy	1700 (1.7K)	1248 (1.2K)
16	War	1648 (1.6K)	1102 (1.1K)
17	Biography	1590 (1.6K)	1173 (1.2K)
18	Noir	1126 (1.1K)	593
19	Documentary	700	643
20	Superhero	612	449
21	Sports	462	423
22	Suspense	351	317

23	Historical	326	287
24	Short	282	103
25	Spy	275	250
26	Satire	230	208
27	Disaster	200	180
28	Teen	162	145
29	Political	93	83
30	Erotic	92	80
31	Live Action	86	79
32	Martial Arts	70	67
33	Supernatural	65	62
34	Dance	59	51
35	Performance	55	54
36	Slasher	51	49
37	Sport	17	17
38	Silent	16	16
39	Legal	10	10
40	Independent	3	3