

Claudia del Pozo Iglesias

# Entrega NoSQL

Diciembre 2023

## Evaluación

### 0. Importación del Json en una colección llamada "movies"

La importación se hizo de forma manual en MongoDBCompass y NoSQLBooster.

### 1. Analizar con find la colección.

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

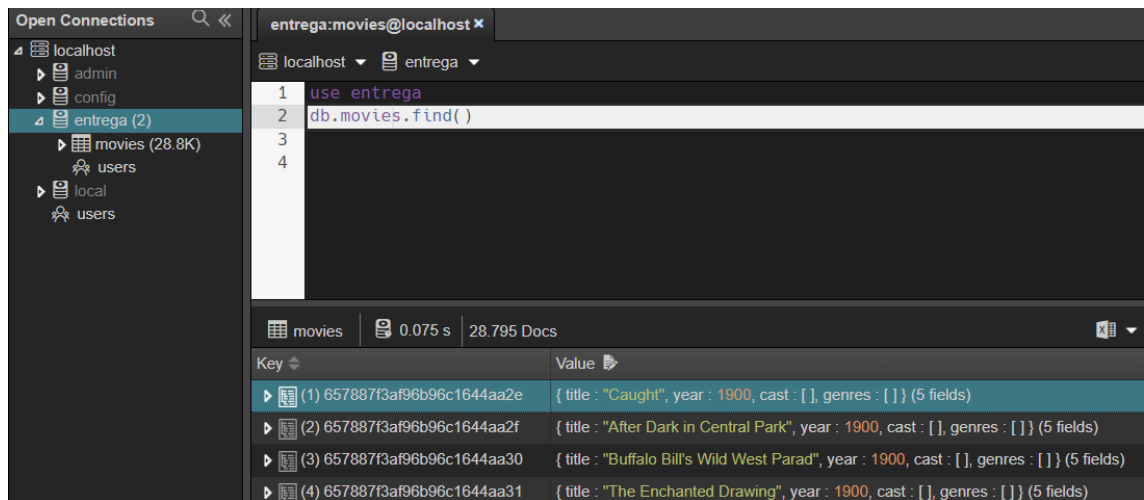


Imagen 1

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

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

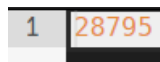


Imagen 2

### 3. Insertar una película.

```
var pelicula = {  
  title: "Barbie", year: 2023,  
  cast: ["Margott Robie", "Ryan Gosling"],  
  genres: ["Comedy", "Fantasy"]  
}  
db.movies.insertOne(pelicula)
```

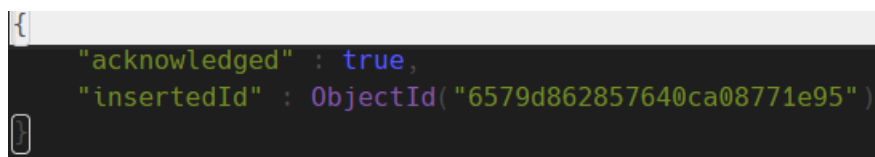


Imagen 3

### 4. Borrar la película insertada en el punto anterior.

```
var titulo = { title: "Barbie" }  
db.movies.deleteMany(titulo)
```

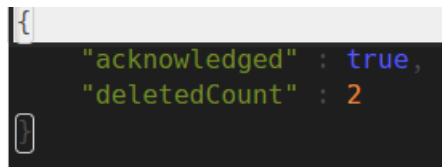


Imagen 4

5. Contar cuantas películas tienen actores (cast) que se llaman "and".

```
var consulta = { cast: "and" }  
db.movies.find(consulta).count()
```



Imagen 5

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 actores.

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

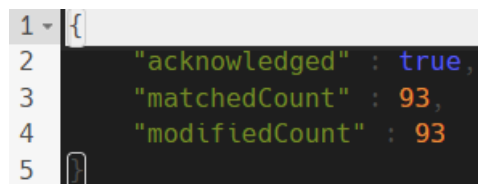


Imagen 6

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

```
var consulta = { cast: [] }  
db.movies.find(consulta).count()
```

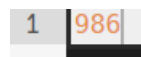


Imagen 7

8. 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"].

```
var query = {cast : []}  
var operacion = { $addToSet: {cast:"Undefined"}}  
db.movies.updateMany(query, operacion)
```

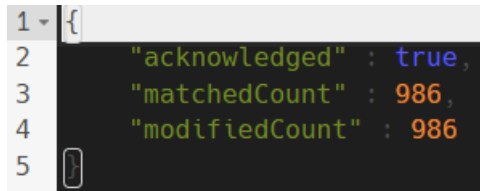


Imagen 8

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

```
var consulta = {genres: []}  
db.movies.find(consulta).count()
```



Imagen 9

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"] .

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



Imagen 10

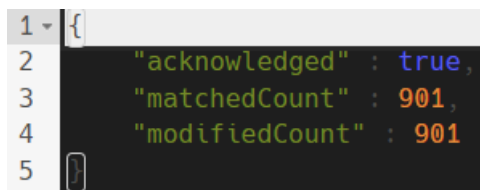


Imagen 11

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

```
db.movies.find({}).sort({year: -1})  
.limit(1)  
.project({"_id": 0, year: 1})
```

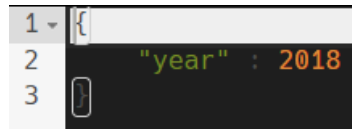


Imagen 12

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.

```
var docu = {title:"$title", year:"$year", cast: "$cast", genres: "$genres"}
var fase1 = {
  $group: {
    "_id": null,
    maxi: {$max: "$year"},
    movies: { $push: docu}
  }
}
var fase2 = {$unwind: "$movies"}
var fase3 = {
  $replaceWith: {
    $mergeObjects:
      [{"maxYear": {$subtract: ["$maxi", 20]}}, "$movies"]
  }
}
var fase4 = {$match: {$expr: {$gte: ["$year", "$maxYear"]}}}
var fase5 = { $group: { "_id": null, peliculas: {$sum: 1}}}
var pipeline = [fase1, fase2, fase3, fase4, fase5]
db.movies.aggregate(pipeline)
```

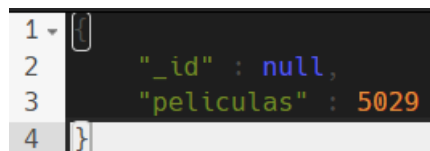


Imagen 13

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.

```
var op1 = {$match: {year: {$gte: 1960, $lte: 1969}}}
var op2 = {$group: { "_id": null, total: {$sum: 1}}}
var pipeline = [op1, op2]
db.movies.aggregate(pipeline)
```

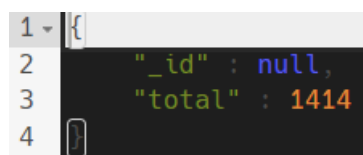
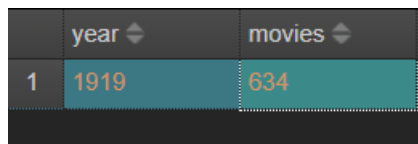


Imagen 14

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 op1 = { $group: { "_id": "$year", movies: { $sum: 1 } } }
var op2 = { $project: { "_id": 0, year: "$_id", movies: "$movies" } }
var op3 = { $group: { "_id": "$movies", years: { $push: "$year" } } }
var op4 = { $project: { "_id": 0, movies: "$_id", years: "$years" } }
var op5 = { $sort: { "movies": -1 } }
var op6 = { $limit: 1 }
var op7 = { $unwind: "$years" }
var op8 = { $project: { year: "$years", movies: "$movies" } }
var pipeline = [op1, op2, op3, op4, op5, op6, op7, op8]
db.movies.aggregate(pipeline)
```

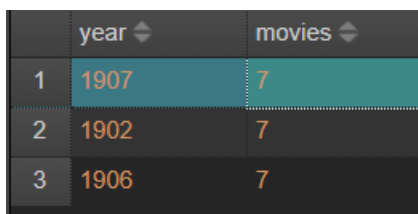


	year	movies
1	1919	634

Imagen 15

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.

```
var op1 = { $group: { "_id": "$year", movies: { $sum: 1 } } }
var op2 = { $project: { "_id": 0, year: "$_id", movies: "$movies" } }
var op3 = { $group: { "_id": "$movies", years: { $push: "$year" } } }
var op4 = { $project: { "_id": 0, movies: "$_id", years: "$years" } }
var op5 = { $sort: { "movies": 1 } }
var op6 = { $limit: 1 }
var op7 = { $unwind: "$years" }
var op8 = { $project: { year: "$years", movies: "$movies" } }
var pipeline = [op1, op2, op3, op4, op5, op6, op7, op8]
db.movies.aggregate(pipeline)
```



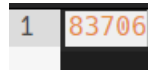
	year	movies
1	1907	7
2	1902	7
3	1906	7

Imagen 16

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

```
var op1 = { $unwind: "$cast" }
var op2 = { $project: { "_id": 0 } }
var op3 = { $out: "actors" }
var ops = [op1, op2, op3]
db.movies.aggregate(ops)
```

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



1	83706
---	-------

Imagen 17

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 op1 = { $match: { cast: { $ne: "Undefined" } } }

var op2 = { $group: { "_id": "$cast", peliculas: { $sum: 1 } } }

var op3 = { $sort: { peliculas: -1 } }

var op4 = { $limit: 5 }

var op5 = {
  $project: {
    "_id": 0,
    "Actor": "$_id",
    "Peliculas": "$peliculas" } }

var pipeline = [op1, op2, op3, op4, op5]

db.actors.aggregate(pipeline)
```



	Actor ↕	Peliculas ↕
1	Harold Lloyd	190
2	Hoot Gibson	142
3	John Wayne	136
4	Charles Starrett	116
5	Bebe Daniels	103

Imagen 18

18. 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.

```
var op1 = { $match: { cast: { $ne: "Undefined" } } }

var op2 = {
  $group: {
    "_id": {
      pelicula: "$title", year: "$year"
    },
    actores: { $addToSet: "$cast" }
  }
}
```

```

    }
  }
  var op3 = { $addFields: {countActors: {$size: "$actores"}}}
  var op4 = { $sort: { countActors: -1 } }
  var op5 = { $limit: 5 }
  var op6 = { $project: {actores: 0} }
  var pipeline = [op1, op2, op3, op4, op5, op6]
  db.actors.aggregate(pipeline)

```

	_id		countActors
	pelicula	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

Imagen 19

19. 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.

```

var op1 = { $match: {cast: { $ne: "Undefined"}}}
var op2 = {
  $group: {
    _id: "$cast",
    inicio: { $min: "$year" },
    fin: { $max: "$year" }
  }
}
var op3 = {
  $addFields: {duracion: {$subtract: ["$fin", "$inicio"]}}
}
var op4 = { $sort: { duracion : -1} }
var op5 = { $limit: 5 }
var op6 = {
  $project: {
    _id: 0,
    Actor: "$_id",
    inicioCarrera: "$inicio",
    finCarrera: "$fin",
    Duracion: "$duracion"
  }
}

var pipeline = [op1, op2, op3, op4, op5, op6]
db.actors.aggregate(pipeline)

```



	Actor ▲	inicioCarrera ⇅	finCarrera ⇅	Duracion ⇅
1	Gloria Stuart	1932 (1.9K)	2012 (2.0K)	80
2	Harrison Ford	1919 (1.9K)	2017 (2.0K)	98
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

Imagen 20

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.

```
var op1 = { $unwind: "$genres" }
var op2 = { $project: { _id: 0 } }
var op3 = {
  $merge: {
    into: "genres",
    whenMatched: "merge",
    whenNotMatched: "insert"
  }
}
```

```
var operaciones = [op1, op2, op3]
db.actors.aggregate(operaciones)
db.genres.find().count()
```

1	210864
---	--------

Imagen 21

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.

```
var op1 = { $match: { genres: { $ne: "Undefined" } } }
var op2 = {
  $group: {
    "_id": {
      genero: "$genres", year: "$year"
    },
    peliculas: { $addToSet: "$title" } } }
var op3 = { $addFields: { countPeliculas: { $size: "$peliculas" } } }
var op4 = { $sort: { countPeliculas: -1 } }
var op5 = { $limit: 5 }
var op6 = { $project: { peliculas : 0 } }
```

```
var pipeline = [op1, op2, op3, op4, op5, op6]
```

```
db.genres.aggregate(pipeline)
```

	_id		countPelículas
	genero	year	
1	Drama	1919	291
2	Drama	1925	247
3	Drama	1924	233
4	Comedy	1919	226
5	Drama	1922	209

Imagen 22

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.

```
var op1 = { $match: { cast: { $ne: "Undefined" } } }
```

```
var op2 = {
```

```
  $group : {
```

```
    "_id": "$cast",
```

```
    genres: { $addToSet: "$genres" }
```

```
  }
```

```
}
```

```
var op3 = { $addFields: { numGeneros: { $size: "$genres" } } }
```

```
var op4 = { $sort: { numGeneros: -1 } }
```

```
var op5 = { $limit: 5 }
```

```
var op6 = {
```

```
  $project: {
```

```
    _id: 0,
```

```
    Actor: "$_id",
```

```
    numGeneros: "$numGeneros",
```

```
    Generos: "$genres"
```

```
  }
```

```
}
```

```
var pipeline = [op1, op2, op3, op4, op5, op6]
```

```
db.genres.aggregate(pipeline)
```

	Actor	numGeneros	Generos
1	Dennis Quaid	20	Array[20]
2	James Mason	19	Array[19]
3	Michael Caine	19	Array[19]
4	Danny Glover	18	Array[18]
5	Johnny Depp	18	Array[18]

Imagen 23

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.

```
var op1 = {
  $group: {
    "_id": {
      title: "$title", year: "$year"
    },
    generos: { $addToSet: "$genres" }
  }
}
var op2 = { $addFields: { numGeneros: { $size: "$generos" } } }
var op3 = { $sort: { numGeneros: -1 } }
var op4 = { $limit: 5 }
var pipeline = [op1, op2, op3, op4]
db.genres.aggregate(pipeline)
```

```
1 /* 1 */
2 {
3   "_id" : {
4     "title" : "American Made",
5     "year" : 2017
6   },
7   "generos" : [
8     "Crime",
9     "Historical",
10    "Drama",
11    "Thriller",
12    "Biography",
13    "Action",
14    "Comedy"
15  ],
16   "numGeneros" : 7
17 },
18 }
```

Imagen 24

Key	Value
▶ (1) { title : "American Made", year : 2017 }	{ generos : [ "Crime", "Historical", "Drama", "Thriller", "Biography", "Action", "Comedy" ], numGeneros : 7 }
▶ (2) { title : "The Dark Tower", year : 2017 }	{ generos : [ "Fantasy", "Action", "Adventure", "Science Fiction", "Horror", "Western" ], numGeneros : 6 }
▶ (3) { title : "My Little Pony: The Movie", year : 2017 }	{ generos : [ "Animated", "Comedy", "Family", "Musical", "Adventure", "Fantasy" ], numGeneros : 6 }
▶ (4) { title : "Thor: Ragnarok", year : 2017 }	{ generos : [ "Science Fiction", "Action", "Adventure", "Superhero", "Comedy", "Fantasy" ], numGeneros : 6 }
▶ (5) { title : "Dunkirk", year : 2017 }	{ generos : [ "War", "Historical", "Thriller", "Drama", "Adventure", "Action" ], numGeneros : 6 }

Imagen 25

	_id		generos	numGeneros
	title	year		
1	American Made	2017	Array[7]	7
2	The Dark Tower	2017	Array[6]	6
3	My Little Pony: The Movie	2017	Array[6]	6
4	Thor: Ragnarok	2017	Array[6]	6
5	Dunkirk	2017	Array[6]	6

Imagen 26

24. Sobre la colección "movies", calcular la cantidad promedio de actores que participan en cada película por año. Mostrar el resultado ordenado por año de manera descendente.

```
var fase1 = { $match: {cast: { $ne: "Undefined" }}}

var fase2 = {
  $group: {
    _id: "$year", actores: { $addToSet: "$cast" }
  }
}

var fase3 = { $addFields: { countActores: { $size: "$actores" } } }
var fase4 = { $sort: { countActores: -1 } }
var fase5 = { $limit: 10 }
var pipeline = [fase1, fase2, fase3, fase4, fase5]
db.actors.aggregate(pipeline)
```

	_id	actores	countActores
1	2012	Array[1816]	1816 (1.8K)
2	2013	Array[1602]	1602 (1.6K)
3	2011	Array[1275]	1275 (1.3K)
4	2017	Array[1266]	1266 (1.3K)
5	2018	Array[1207]	1207 (1.2K)
6	2010	Array[1062]	1062 (1.1K)
7	2006	Array[983]	983
8	2009	Array[976]	976
9	1996	Array[825]	825
10	2007	Array[794]	794

Imagen 27

25. Mostrar las 5 décadas con mayor cantidad de películas publicadas.

```
var fase1 = {
  $addFields: { decada: { $subtract: ["$year", { $mod: ["$year", 10] } ] } }
}

var fase2 = { $group: { "_id": "$decada", peliculas: { $sum: 1 } } }
var fase3 = { $sort: { peliculas: -1 } }
var fase4 = { $limit: 5 }
```

```

var fase5 = {
  $project : { _id : 0, Decada : "$_id", Peliculas : "$peliculas" }
}
var pipeline = [fase1 , fase2 , fase3 , fase4 , fase5 ]
db.movies.aggregate ( pipeline )

```

	Decada ↕	Peliculas ↕
1	1930 (1.9K)	4104 (4.1K)
2	1940 (1.9K)	4023 (4.0K)
3	1920 (1.9K)	3584 (3.6K)
4	1950 (1.9K)	3252 (3.3K)
5	1990 (2.0K)	2686 (2.7K)

Imagen 28

26. Obtener la lista de los 5 actores que entre los años 1995 y 1999 hayan hecho entre 5 y 10 películas.

```

var fase1 = { $match: { cast: { $ne: "Undefined" }, year: { $gte: 1995, $lte: 1999 } } }
var fase2 = { $unwind: "$cast" }
var fase3 = {
  $group: {
    _id: "$cast",
    peliculas: { $sum: 1 } }
}
var fase4 = {
  $match: { peliculas: { $gte: 5, $lte: 10 } } }
var fase5 = {
  $project: {
    _id: 0,
    Actor: "$_id",
    TotalPeliculas: "$peliculas" }
}
var fase6 = { $sort: { TotalPeliculas: 1 } }
var fase7 = { $limit : 5 }
var pipeline = [fase1 , fase2 , fase3 , fase4 , fase5 , fase6 , fase7 ]
db.actors.aggregate ( pipeline )

```

	Actor ↕	TotalPeliculas ↕
1	Arliss Howard	5
2	Sherilyn Fenn	5
3	Philip Baker Hall	5
4	Annette Bening	5
5	Ellen Barkin	5

Imagen 29