

Proyecto de Base de datos en Mongo
DB.

Películas Walt Disney



Asunción de los Ángeles Naranjo Rodríguez
30 de Abril del 2024



Introducción

1. Título
2. Autor
3. Ilustrador
4. Ficción o no ficción
5. ¿Por qué elegiste este libro?

1. Diseño de la base de datos

La base de datos con la que vamos a trabajar llamada “películas disney” está integrada por tres colecciones, una llamada películas, otra llamada personajes, y por último, una llamada canciones.

Dentro de la colección películas se van a introducir datos relacionados con películas de Disney que poseen la siguiente estructura:

- Título
- Año de lanzamiento
- Década
- Género
- Duración
- Director
- Descripción

Dentro de la colección personajes se van a introducir datos relacionados con alguno de los personajes de Disney que actúan en las películas, quedando una estructura como la siguiente:

- Nombre del personaje
- Función en la película
- Descripción

Dentro de la colección canciones se van a introducir datos relacionados con algunas de las canciones que se convirtieron en banda sonora de las películas de Disney que actúan, quedando una estructura como la siguiente:

- Título de la canción
- Película en la que aparece
- Personajes que interpretan las canciones

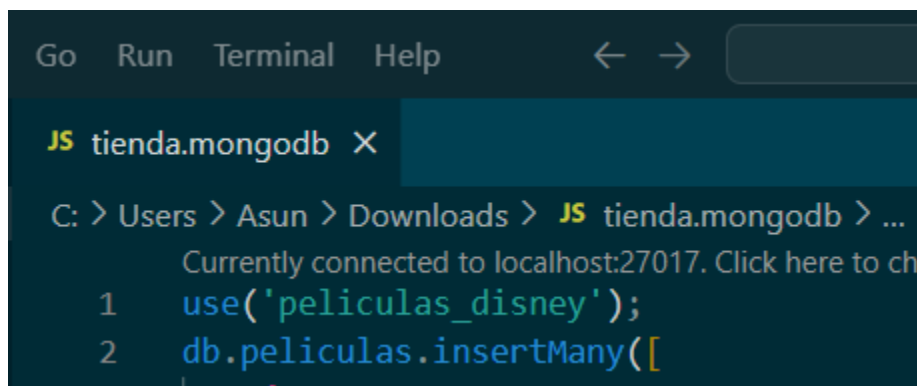
2. Creación de la base de datos y diseño de las colecciones.

Para la creación de la base de datos se han dado una serie de pasos que se describen a continuación.

PRIMER PASO: creamos en nuestro visual studio code un documento JSON al que vamos a denominar peliculas.mongodb.

SEGUNDO PASO: una vez que hemos abierto el documento, escribimos las siguientes instrucciones:

- “Use” seguido entre paréntesis del nombre que le vamos a poner a nuestra base de datos, en nuestro caso, ‘peliculas_disney’.
- “db.peliculas.insertMany ()” seguido de la estructura JSON que va a tener nuestra nuestra colección.



```
Go Run Terminal Help
JS tienda.mongodb X
C: > Users > Asun > Downloads > JS tienda.mongodb > ...
Currently connected to localhost:27017. Click here to ch
1 use('peliculas_disney');
2 db.peliculas.insertMany([
```

Se ejecuta tres veces la instrucción “insert.Many ()”, uno por cada una de las colecciones que se vaya a implementar en nuestra base de datos. En este caso, se han creado tres colecciones llamadas canciones, películas y personajes.



```
Go Run Terminal Help
JS tienda.mongodb X JS consultasProyecto.mongodb
C: > Users > Asun > Downloads > JS tienda.mongodb > "valoracion"
Currently connected to localhost:27017. Click here to change connection.
1 use('peliculas_disney');
2 db.peliculas.insertMany([
3 {
4   "titulo": "La sirenita",
5   "añoLanzamiento": 1989,
6   "decada": "Década de los 80",
7   "genero": "Animación",
8   "duracion": 79,
9   "director": ["John Musker", "Ron Clements"],
10  "descripcion": "Ariel, hija del rey Tritón, es la princesa de las sirenas. Está a punto de celebrarse su fiesta de cumpleaños y su mayor ilu
11  "valoracion": 5
12 },
204 db.personajes.insertMany([
205 {
206   "nombre": "Ariel",
207   "funcion": "Salir a la superficie para explorar el mundo humano",
208   "descripcion": "Ariel es la protagonista de la película y es una joven sirena que anhela explorar el mundo de los humanos. Se siente atraída
209   "titulo": "La sirenita"
210 },
```

```
db.canciones.insertMany([
  {
    "titulo": "Bajo el mar",
    "pelicula": "La sirenita",
    "interpretes": ["Sebastian", "Ariel"],
    "reproducciones": 78000213
  },

```

TERCER PASO: En cada una de las colecciones que se han creado se introducen los datos que se corresponden con cada campo que hemos establecido previamente.

La colección películas posee la siguiente estructura:

```
{
  "titulo": "Aladdin",
  "añoLanzamiento": 1992,
  "decada": "Década de los 90",
  "genero": "Animación",
  "duracion": 87,
  "director": ["John Musker", "Ron Clements"],
  "descripcion": "Basada en el famoso cuento 'Aladino y la lámpara maravillosa', la trama se sitúa en el exótico paisaje del mítico reino árabe",
  "valoracion": 3
},
```

La colección personajes posee la siguiente estructura:

```
{
  "nombre": "Conejo Blanco",
  "funcion": "Llevar a Alicia al país de las maravillas",
  "descripcion": "Es el primer personaje que Alicia encuentra en el País de las Maravillas. Es nervioso, obsesionado con el tiempo y a menudo p",
  "titulo": "Alicia en el país de las maravillas"
},
```

La colección canciones posee la siguiente estructura:

```
{
  "titulo": "Todos quieren ser un gato Jazz",
  "pelicula": "Los aristogatos",
  "interpretes": ["Thomas O'Malley", "Gatos callejeros"],
  "reproducciones": 33123564
},
```

CUARTO PASO: una vez que se han introducido todos los datos con los que se va a trabajar, se clicla sobre el botón de reproducir y ya está creada la base de datos. con la que se va a trabajar.

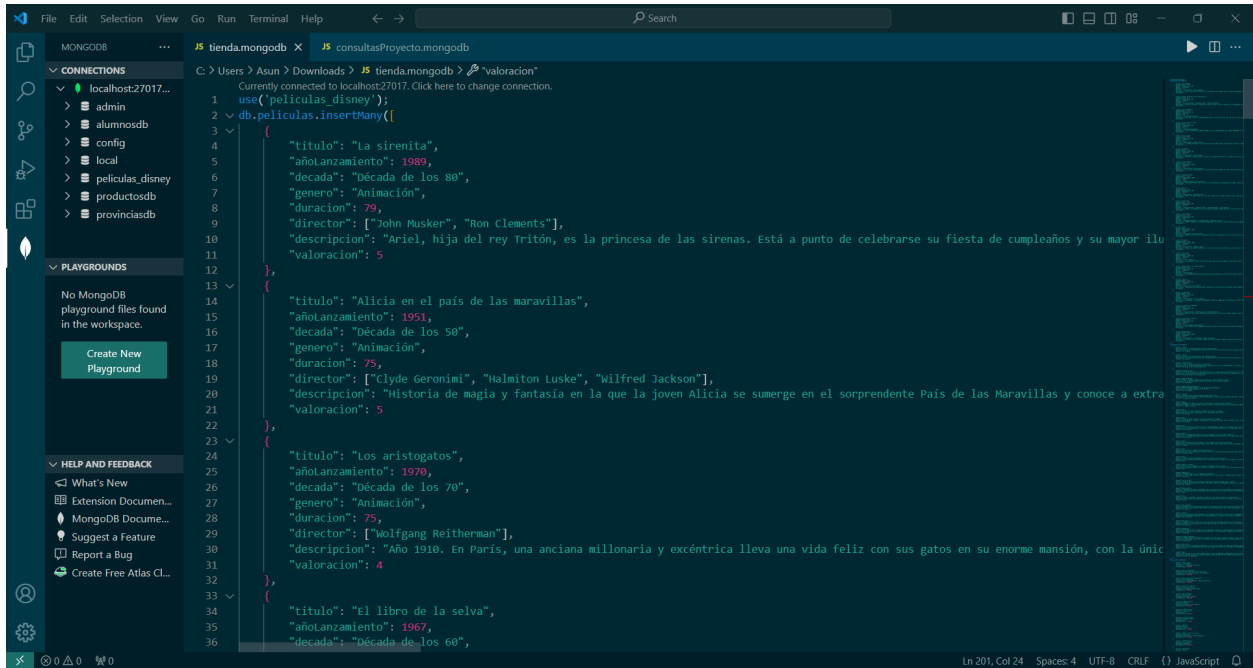
```
1 use('peliculas_disney');
2 db.peliculas.insertMany([
3   {
4     "titulo": "La sirenita",
5     "añoLanzamiento": 1989,
6     "decada": "Década de los 80",
7     "genero": "Animación",
8     "duracion": 79,
9     "director": ["John Musker", "Ron Clements"],
10    "descripcion": "Ariel, hija del rey Tritón, es la princesa",
11    "valoracion": 5
12  },
13  {
14    "titulo": "Alicia en el país de las maravillas",
15    "añoLanzamiento": 1951,
16    "decada": "Década de los 50",
17    "genero": "Animación",
18    "duracion": 75,
19    "director": ["Clyde Geronimi", "Halmiton Luske", "Wilfred"],
20    "descripcion": "Historia de magia y fantasía en la que la",
21    "valoracion": 5
22  },
23  {
24    "titulo": "Los aristogatos",
25    "añoLanzamiento": 1970,
26    "decada": "Década de los 70",
27    "genero": "Animación",
28    "duracion": 75,
29    "director": ["Wolfgang Reitherman"],
30    "descripcion": "Año 1910. En París, una anciana millonaria",
31    "valoracion": 4
32  },
33  {
34    "titulo": "El libro de la selva",
35    "añoLanzamiento": 1967,
36    "decada": "Década de los 60",
```

Collection Name	Storage size	Documents	Avg. document size	Indexes	Total index size
canciones	4.10 kB	19	127.00 B	1	4.10 kB
peliculas	4.10 kB	20	636.00 B	1	4.10 kB
personajes	4.10 kB	27	328.00 B	1	4.10 kB

3. Carga de datos.

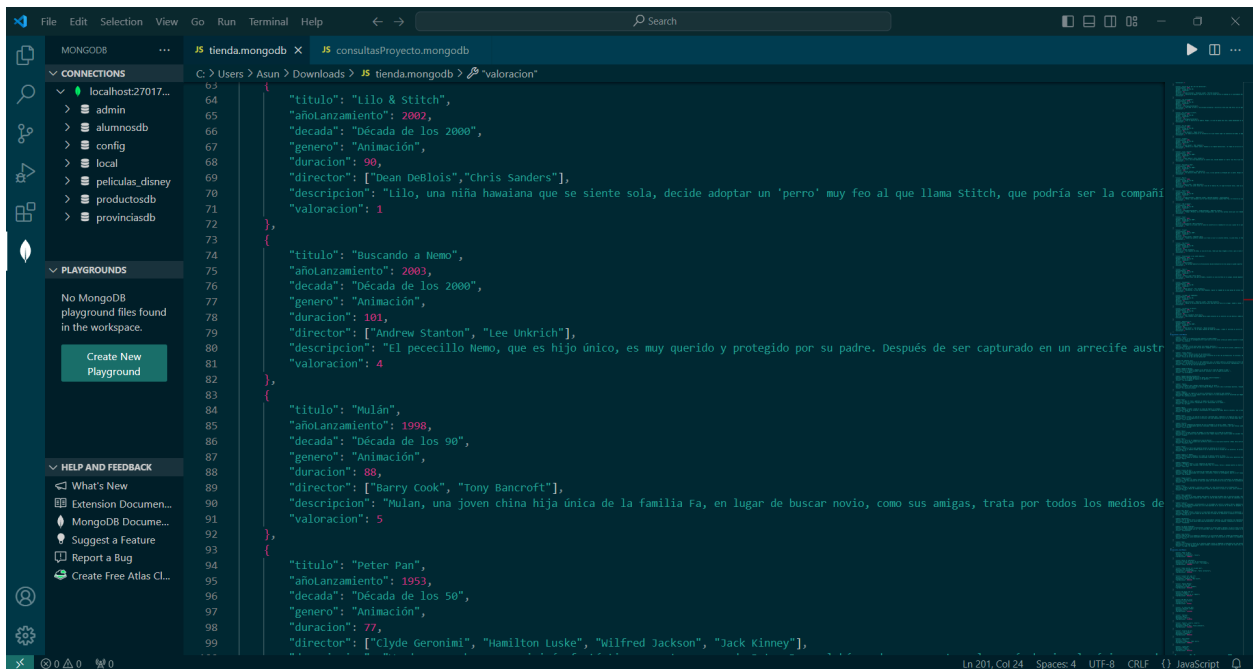
Como se ha mencionado en el apartado anterior, para cargar los datos se realizan insertManys de cada una de las colecciones, de esta forma, se generan los datos y quedan almacenados en la base de datos con la que vamos a trabajar.

PELÍCULAS



The screenshot shows the VS Code interface with a MongoDB playground. The left sidebar shows the 'CONNECTIONS' panel with 'localhost:27017' selected. The 'PLAYGROUNDS' panel shows 'No MongoDB playground files found in the workspace.' The main editor displays a JavaScript script for inserting movie data into a MongoDB collection named 'peliculas'.

```
1 use('peliculas_disney');
2 db.peliculas.insertMany([
3   {
4     "titulo": "La sirenita",
5     "añoLanzamiento": 1989,
6     "decada": "Década de los 80",
7     "genero": "Animación",
8     "duracion": 79,
9     "director": ["John Musker", "Ron Clements"],
10    "descripcion": "Ariel, hija del rey Tritón, es la princesa de las sirenas. Está a punto de celebrarse su fiesta de cumpleaños y su mayor ilusión es casarse con el príncipe Eric.",
11    "valoracion": 5
12  },
13  {
14    "titulo": "Alicia en el país de las maravillas",
15    "añoLanzamiento": 1951,
16    "decada": "Década de los 50",
17    "genero": "Animación",
18    "duracion": 75,
19    "director": ["Clyde Geronimi", "Halmiton Luske", "Wilfred Jackson"],
20    "descripcion": "Historia de magia y fantasía en la que la joven Alicia se sumerge en el sorprendente País de las Maravillas y conoce a extraños personajes.",
21    "valoracion": 5
22  },
23  {
24    "titulo": "Los aristogatos",
25    "añoLanzamiento": 1970,
26    "decada": "Década de los 70",
27    "genero": "Animación",
28    "duracion": 75,
29    "director": ["Wolfgang Reitherman"],
30    "descripcion": "Año 1910. En París, una anciana millonaria y excéntrica lleva una vida feliz con sus gatos en su enorme mansión, con la única condición: que no se vayan.",
31    "valoracion": 4
32  },
33  {
34    "titulo": "El libro de la selva",
35    "añoLanzamiento": 1967,
36    "decada": "Década de los 60",
```



The screenshot shows the VS Code interface with a MongoDB playground. The left sidebar shows the 'CONNECTIONS' panel with 'localhost:27017' selected. The 'PLAYGROUNDS' panel shows 'No MongoDB playground files found in the workspace.' The main editor displays a JavaScript script for inserting movie data into a MongoDB collection named 'peliculas'.

```
64 "titulo": "Lilo & Stitch",
65 "añoLanzamiento": 2002,
66 "decada": "Década de los 2000",
67 "genero": "Animación",
68 "duracion": 90,
69 "director": ["Dean DeBlois", "Chris Sanders"],
70 "descripcion": "Lilo, una niña hawaiana que se siente sola, decide adoptar un 'perro' muy feo al que llama Stitch, que podría ser la compañía perfecta.",
71 "valoracion": 1
72 },
73 {
74   "titulo": "Buscando a Nemo",
75   "añoLanzamiento": 2003,
76   "decada": "Década de los 2000",
77   "genero": "Animación",
78   "duracion": 101,
79   "director": ["Andrew Stanton", "Lee Unkrich"],
80   "descripcion": "El pececillo Nemo, que es hijo único, es muy querido y protegido por su padre. Después de ser capturado en un arrecife australiano, su padre emprende un viaje para rescatarlo.",
81   "valoracion": 4
82 },
83 {
84   "titulo": "Mulán",
85   "añoLanzamiento": 1998,
86   "decada": "Década de los 90",
87   "genero": "Animación",
88   "duracion": 88,
89   "director": ["Barry Cook", "Tony Bancroft"],
90   "descripcion": "Mulán, una joven china hija única de la familia Fa, en lugar de buscar novio, como sus amigas, trata por todos los medios de convertirse en soldado.",
91   "valoracion": 5
92 },
93 {
94   "titulo": "Peter Pan",
95   "añoLanzamiento": 1953,
96   "decada": "Década de los 50",
97   "genero": "Animación",
98   "duracion": 77,
99   "director": ["Clyde Geronimi", "Hamilton Luske", "Wilfred Jackson", "Jack Kinney"],
```

PERSONAJES

```
db.personajes.insertMany([
  {
    "nombre": "Ariel",
    "funcion": "Salir a la superficie para explorar el mundo humano",
    "descripcion": "Ariel es la protagonista de la película y es una joven sirena que anhela explorar el mundo de los humanos. Se siente atraída",
    "titulo": "La sirenita"
  },
  {
    "nombre": "Ursula",
    "funcion": "Manipula a Ariel para lograr sus propios objetivos.",
    "descripcion": "Es una bruja del mar que ansia venganza contra el reino del mar. Es astuta, poderosa y malvada.",
    "titulo": "La sirenita"
  },
  {
    "nombre": "Conejo Blanco",
    "funcion": "Llevar a Alicia al país de las maravillas",
    "descripcion": "Es el primer personaje que Alicia encuentra en el País de las Maravillas. Es nervioso, obsesionado con el tiempo y a menudo p",
    "titulo": "Alicia en el país de las maravillas"
  },
  {
    "nombre": "El sombrero loco",
    "funcion": "Ser un anfitrión en el té del Sombrero Loco, un evento caótico y surrealista en el País de las Maravillas",
    "descripcion": "Personaje excéntrico que Alicia encuentra en su viaje. Es excéntrico, impredecible y a menudo parece vivir en su propio mund",
    "titulo": "Alicia en el país de las maravillas"
  },
  {
    "nombre": "Thomas O'Malley",
    "funcion": "Guiar y proteger a Duquesa y sus gatitos en su viaje de regreso a casa.",
    "descripcion": "Es un gato callejero simpático, valiente y con un toque de picardía.",
    "titulo": "Los aristogatos"
  },
  {
    "nombre": "Madame Adelaide Bonfamille",
    "funcion": "Mujer adinerada amante de los gatos que cuida de duquesa.",
    "descripcion": "Es la dueña de duquesa y los gatitos",
    "titulo": "Los aristogatos"
  }
])
```

```
{
  "nombre": "Capitán Li Shang",
  "funcion": "Entrena a los soldados y los lidera en la batalla contra los hunos.",
  "descripcion": "Li Shang es valiente, disciplinado y honorable. A lo largo de la película, desarrolla una estrecha relación con Mulán, sin sa",
  "titulo": "Mulán"
},
{
  "nombre": "Campanilla",
  "funcion": "Protege a Peter y a sus compañeros de aventuras",
  "descripcion": "Es el hada traviesa y leal de Peter Pan. Es celosa, terca y a menudo muestra un afecto especial por Peter. Aunque no puede h",
  "titulo": "Peter Pan"
},
{
  "nombre": "Cruella de Vil",
  "funcion": "Intenta robar a los cachorros de dalmata para hacerse un abrigo con sus pieles.",
  "descripcion": "Mujer obsesionada con la moda y las pieles. Cruella es extravagante, egoísta y está dispuesta a hacer cualquier cosa para con",
  "titulo": "101 dalmatas"
},
{
  "nombre": "Miguel Rivera",
  "funcion": "Perseguir sus sueños relacionados con su pasión por la música y descubrir la verdad sobre su ancestro familiar.",
  "descripcion": "Un niño mexicano de 12 años que sueña con convertirse en músico, a pesar de la prohibición de la música en su familia. Es val",
  "titulo": "Coco"
},
{
  "nombre": "Elsa",
  "funcion": "Aceptar y aprender a controlar su poder, así como reconciliarse con su hermana Anna.",
  "descripcion": "Es la Reina de Arendelle. Tiene el poder de controlar el hielo y la nieve, pero lucha por controlarlo debido al miedo a herir",
  "titulo": "Frozen"
},
{
  "nombre": "Buzz Lightyear",
  "funcion": "Pretende ser el nuevo juguete que desafía la posición de Woody como el favorito de Andy y enseñarle a Woody lecciones importantes",
  "descripcion": "Es un juguete espacial que llega a la vida de Andy como un nuevo regalo de cumpleaños. Al principio, cree ser un verdadero gu",
  "titulo": "Toy Story"
},
{
  "nombre": "Woody",
  "funcion": "Es el juguete favorito de Andy y el líder del grupo de juguetes.",
  "descripcion": "Es un vaquero de juguete que ha sido el favorito de Andy durante años. Es leal, valiente y siempre está dispuesto a ayudar a sus amigos.",
  "titulo": "Toy Story"
}
```

CANCIONES

The screenshot shows the VS Code editor with a MongoDB query in the console. The query is a `db.canciones.insertMany()` command. The left sidebar shows the 'CONNECTIONS' panel with a list of databases: admin, alumnosdb, config, local, peliculas_disney, productosdb, and provinciasdb. The 'PLAYGROUNDS' panel shows 'No MongoDB playground files found in the workspace.' The 'HELP AND FEEDBACK' panel shows 'What's New', 'Extension Document...', 'MongoDB Document...', 'Suggest a Feature', 'Report a Bug', and 'Create Free Atlas CL...'. The console output shows the following JSON array:

```
db.canciones.insertMany([
  {
    "titulo": "Bajo el mar",
    "pelicula": "La sirenita",
    "interpretes": ["Sebastian", "Ariel"],
    "reproducciones": 78000213
  },
  {
    "titulo": "Feliz no cumpleaños",
    "pelicula": "Alicia en el país de las maravillas",
    "interpretes": ["El sombrero loco", "sus amigos"],
    "reproducciones": 45030987
  },
  {
    "titulo": "Todos quieren ser un gato Jazz",
    "pelicula": "Los aristogatos",
    "interpretes": ["Thomas O'Malley", "Gatos callejeros"],
    "reproducciones": 33123564
  },
  {
    "titulo": "¿Quiero ser como tú?",
    "pelicula": "Libro de la Selva",
    "interpretes": ["Mowgli", "Rey Louie"],
    "reproducciones": 85127883
  },
  {
    "titulo": "Hakuna Matata",
    "pelicula": "El rey León",
    "interpretes": ["Timón", "Pumba"],
    "reproducciones": 91430298
  },
  {
    "titulo": "Un amigo como yo",
    "pelicula": "Aladdin",
    "interpretes": ["Genio de la lámpara"],
    "reproducciones": 66112114
  }
])
```

The screenshot shows the VS Code editor with a MongoDB query in the console. The query is a `db.canciones.insertMany()` command. The left sidebar shows the 'CONNECTIONS' panel with a list of databases: admin, alumnosdb, config, local, peliculas_disney, productosdb, and provinciasdb. The 'PLAYGROUNDS' panel shows 'No MongoDB playground files found in the workspace.' The 'HELP AND FEEDBACK' panel shows 'What's New', 'Extension Document...', 'MongoDB Document...', 'Suggest a Feature', 'Report a Bug', and 'Create Free Atlas CL...'. The console output shows the following JSON array:

```
db.canciones.insertMany([
  {
    "titulo": "A través del mar",
    "pelicula": "Buscando a Nemo",
    "personajes": ["", ""],
    "reproducciones": 21453663
  },
  {
    "titulo": "Reflejo",
    "pelicula": "Mulán",
    "personajes": ["Mulán"],
    "reproducciones": 47130761
  },
  {
    "titulo": "Volarás, volarás, volarás",
    "pelicula": "Peter Pan",
    "personajes": ["Peter Pan", "Niños perdidos"],
    "reproducciones": 72221654
  },
  {
    "titulo": "Cruella de Vil",
    "pelicula": "101 dálmatas",
    "personajes": ["Roger"],
    "reproducciones": 89013454
  },
  {
    "titulo": "Recuérdame",
    "pelicula": "Coco",
    "personajes": ["Miguel", "Mamá Coco"],
    "reproducciones": 78000213
  },
  {
    "titulo": "Libre soy",
    "pelicula": "Frozen",
    "personajes": ["Elsa"],
    "reproducciones": 68757243
  }
])
```

4. Búsquedas

- Buscar todas las películas de Disney del género "animación".

The screenshot shows the VS Code interface with the MongoDB Playground extension. The left sidebar shows the 'CONNECTIONS' and 'PLAYGROUNDS' panels. The main editor displays a JavaScript file named 'consultasProyecto.mongodb.js' with the following code:

```
1 use("películas_disney");
2
3 //Búsquedas
4 //Todas las películas disney de animación.
5 db.películas.find({
6   "genero": "Animación"
7 })
8
9 //Películas lanzadas después del 2000
10 /*db.películas.find({
11   "añoLanzamiento": { $gt: 2000 }
12 })*/
13
14 //Películas dirigidas por un director específico
15 /*db.películas.find({
16   "director": { $eq: "Wolfgang Reitherman" }
17 })*/
18
19 //Encontrar los personajes principales de una película
20 /*db.personajes.find({
21   $and: [
22     {"titulo": "Mulán"},
23     {"nombre": "Mushu"}
24   ]
25 })*/
26
27 //Buscar canciones Disney interpretadas por un personaje específico
28 /*db.canciones.find({
29   $and: [{"personajes": "Miguel"}, {"personajes": "Mamá Coco"}]
30 })*/
31
32 //Obtener una lista de un director de una de las películas y una década concreta
33 /*db.películas.find({
34   $and: [
35     { $or: [{"director": "Lee Unkrich"}] },
36     { "década": "Década de los 2000" }
37   ]
38 })*/
```

The 'PLAYGROUND RESULT' panel on the right shows the results of the query, displaying two JSON documents for Disney animation movies: 'La sirenita' and 'Alicia en el país de las maravillas'.

- Encontrar películas de Disney lanzadas después de 2000.

The screenshot shows the VS Code interface with the MongoDB Playground extension. The left sidebar shows the 'CONNECTIONS' and 'PLAYGROUNDS' panels. The main editor displays a JavaScript file named 'consultasProyecto.mongodb.js' with the following code:

```
1 use("películas_disney");
2
3 //Búsquedas
4 //Todas las películas disney de animación.
5 db.películas.find({
6   "genero": "Animación"
7 })
8
9 //Películas lanzadas después del 2000
10 db.películas.find({
11   "añoLanzamiento": { $gt: 2000 }
12 })
13
14 //Películas dirigidas por un director específico
15 /*db.películas.find({
16   "director": { $eq: "Wolfgang Reitherman" }
17 })*/
18
19 //Encontrar los personajes principales de una película
20 /*db.personajes.find({
21   $and: [
22     {"titulo": "Mulán"},
23     {"nombre": "Mushu"}
24   ]
25 })*/
26
27 //Buscar canciones Disney interpretadas por un personaje específico
28 /*db.canciones.find({
29   $and: [{"personajes": "Miguel"}, {"personajes": "Mamá Coco"}]
30 })*/
31
32 //Obtener una lista de un director de una de las películas y una década concreta
33 /*db.películas.find({
34   $and: [
35     { $or: [{"director": "Lee Unkrich"}] },
36     { "década": "Década de los 2000" }
37   ]
38 })*/
39
40 //Buscar películas de Disney dirigidas por un director específico
41 /*db.películas.find({
42   "director": { $eq: "Wolfgang Reitherman" }
43 })*/
```

The 'PLAYGROUND RESULT' panel on the right shows the results of the query, displaying two JSON documents for Disney movies released after 2000: 'Lilo & Stitch' and 'Buscando a Nemo'.

- Buscar películas de Disney dirigidas por un director específico.

```
1 use("películas_disney");
2
3 //Búsquedas
4 //Todas las películas disney de animación.
5 > /*db.películas.find({
6
7 //Películas lanzadas después del 2000
8 > /*db.películas.find({
9
10 //Películas dirigidas por un director específico
11 db.películas.find({
12   "director": { $eq: "Wolfgang Reitherman" }
13 })
14
15 //Encontrar los personajes principales de una película
16 db.personajes.find({
17   $and: [
18     { "titulo": "Mulán" },
19     { "nombre": "Mushu" }
20 ] })
21
22 //Buscar canciones Disney interpretadas por un personaje específico
23 db.canciones.find({
24   $and: [{ "personajes": "Miguel" }, { "personajes": "Mamá Coco" } ] })
25
26 //Obtener una lista de un director de una de las películas y una década
27 db.películas.find({
28   $and: [
29     { $or: [{ "director": "Lee Unkrich" } ] },
30     { "década": "Década de los 2000" }
31 ] },
32   .sort({ "titulo": 1 })
33 )
34
35 //Agregaciones
36 db.películas.aggregate([
37   { $group: { _id: "$director", count: { $sum: 1 } } },
38   { $sort: { count: -1 } }
39 ])
40
```

Playground Result

```
1 {
2   "_id": {
3     "$oid": "6649e8f37e1ff1f58205bee9"
4   },
5   "titulo": "Los aristogatos",
6   "añoLanzamiento": 1970,
7   "década": "Década de los 70",
8   "genero": "Animación",
9   "duracion": 75,
10  "director": [
11    "Wolfgang Reitherman"
12  ],
13  "descripcion": "Año 1910. En París, una anciana millonaria y ex
14  "valoracion": 4
15 },
16 {
17   "_id": {
18     "$oid": "6649e8f37e1ff1f58205bee9"
19   },
20   "titulo": "El libro de la selva",
21   "añoLanzamiento": 1967,
22   "década": "Década de los 60",
23   "genero": "Animación",
24   "duracion": 78,
25   "director": [
26     "Wolfgang Reitherman"
27   ],
28   "descripcion": "Tras la muerte de sus padres, Mowgli, un niño d
29   "valoracion": 3
30 },
31 {
32   "_id": {
33     "$oid": "6649e8f37e1ff1f58205bef1"
34   },
35   "titulo": "101 dálmatas",
36   "añoLanzamiento": 1961,
37   "década": "Década de los 60",
38 }
```

- Encontrar personajes principales de una película específica.

```
1 use("películas_disney");
2
3 //Búsquedas
4 //Todas las películas disney de animación.
5 > /*db.películas.find({
6
7 //Películas lanzadas después del 2000
8 > /*db.películas.find({
9
10 //Películas dirigidas por un director específico
11 db.películas.find({
12   "director": { $eq: "Wolfgang Reitherman" }
13 })
14
15 //Encontrar los personajes principales de una película
16 db.personajes.find({
17   $and: [
18     { "titulo": "Mulán" },
19     { "nombre": "Mushu" }
20 ] })
21
22 //Buscar canciones Disney interpretadas por un personaje específico
23 db.canciones.find({
24   $and: [{ "personajes": "Miguel" }, { "personajes": "Mamá Coco" } ] })
25
26 //Obtener una lista de un director de una de las películas y una década
27 db.películas.find({
28   $and: [
29     { $or: [{ "director": "Lee Unkrich" } ] },
30     { "década": "Década de los 2000" }
31 ] },
32   .sort({ "titulo": 1 })
33 )
34
35 //Agregaciones
36 db.películas.aggregate([
37   { $group: { _id: "$director", count: { $sum: 1 } } },
38   { $sort: { count: -1 } }
39 ])
40
41 //Numero total de películas Disney en la base de datos
42 db.películas.aggregate([
43   { $group: { _id: "Total", count: { $sum: 1 } } },
44   { $project: { _id: 0, count: "$count" } }
45 ])
46
```

Playground Result

```
1 {
2   "_id": {
3     "$oid": "6649e8f37e1ff1f58205bf09"
4   },
5   "nombre": "Mushu",
6   "funcion": "Asesora a Mulán para ayudarla durante su estancia e
7   "descripcion": "Es un pequeño dragón guardián de la familia de
8   "titulo": "Mulán"
9 }
10
11
```

- Buscar canciones de Disney interpretadas por un personaje específico.

The screenshot shows the MongoDB Playground interface in VS Code. The left sidebar displays the 'CONNECTIONS' and 'PLAYGROUNDS' sections. The main editor shows a JavaScript file named 'consultasProyecto.mongoddb' with a query for the 'Coco' movie. The 'Playground Result' pane on the right shows the JSON output for the movie 'Coco', including its ID, title, release year, genre, duration, director, and production count.

```
1 use("películas_disney");
2
3
4
5
6
7
8
9 //Buscar canciones Disney interpretadas por un personaje específico
10 db.canciones.find(
11   $and: [{"personajes": "Miguel"}, {"personajes": "Mamá Coco"}]
12 )
13
14
15 //Obtener una lista de un director de una las películas y una década
16 db.películas.find(
17   $and: [
18     { $or: [{"director": "Lee Unkrich"}] },
19     { "década": "Década de los 2000" }
20   ],
21   .sort({ "titulo": 1 })
22 )
23
24 //Agregaciones
25 //Numero total de películas Disney en la base de datos
26 db.películas.aggregate(
27   { $count: "totalPelículas" }
28 )
```

```
1 [
2   {
3     "_id": {
4       "$oid": "6649e8f37e1ff1f58205bf21"
5     },
6     "titulo": "Recuérdame",
7     "película": "Coco",
8     "personajes": [
9       "Miguel",
10      "Mamá Coco"
11    ],
12     "reproducciones": 78000213
13   }
14 ]
```

- Añade una consulta personalizada.

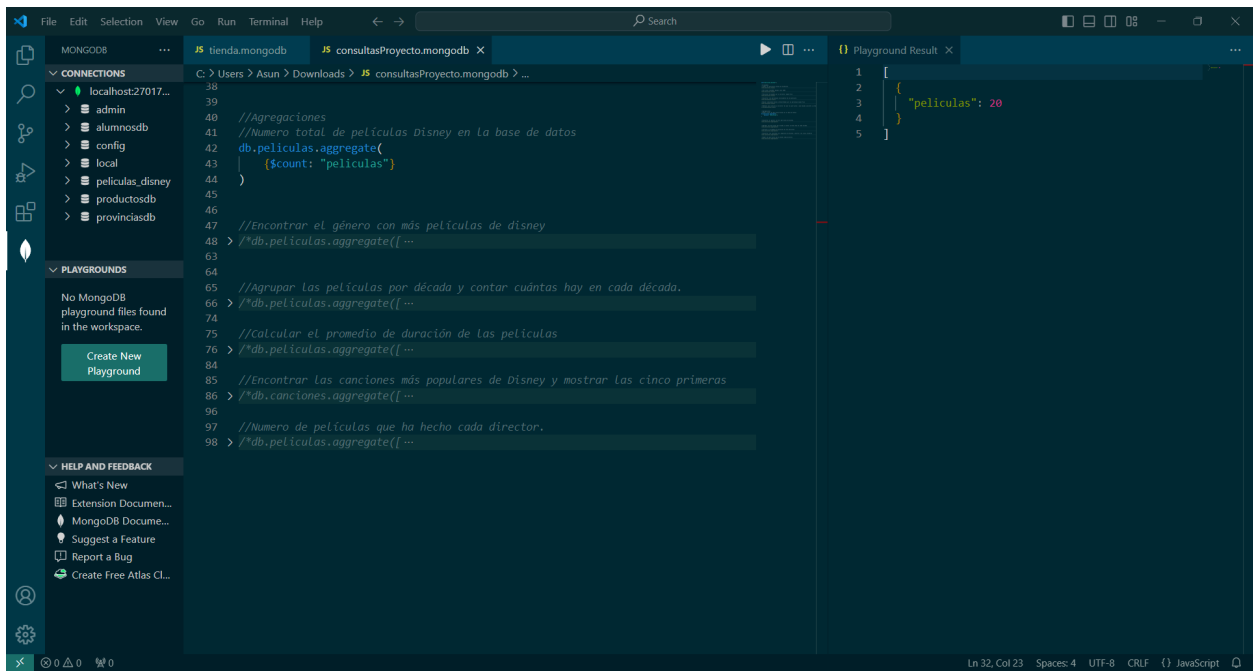
The screenshot shows the MongoDB Playground interface in VS Code. The left sidebar displays the 'CONNECTIONS' and 'PLAYGROUNDS' sections. The main editor shows a JavaScript file named 'consultasProyecto.mongoddb' with a query for the 'Coco' movie. The 'Playground Result' pane on the right shows the JSON output for the movie 'Coco', including its ID, title, release year, genre, duration, director, and production count.

```
1 use("películas_disney");
2
3
4
5
6
7
8
9 //Buscar canciones Disney interpretadas por un personaje específico
10 db.canciones.find(
11   $and: [{"personajes": "Miguel"}, {"personajes": "Mamá Coco"}]
12 )
13
14
15 //Obtener una lista de un director de una las películas y una década
16 db.películas.find(
17   $and: [
18     { $or: [{"director": "Lee Unkrich"}] },
19     { "década": "Década de los 2000" }
20   ],
21   .sort({ "titulo": 1 })
22 )
23
24 //Agregaciones
25 //Numero total de películas Disney en la base de datos
26 db.películas.aggregate(
27   { $count: "totalPelículas" }
28 )
29
30 //Encontrar el género con más películas de disney
31 db.películas.aggregate([
32   { $group: {
33     _id: "$decada",
34     totalPelículas: { $sum: 1 }
35   } }
36 ])
37
38 //Agrupar las películas por década y contar cuántas hay en cada década
39 db.películas.aggregate([
40   { $group: {
41     _id: "$decada",
42     totalPelículas: { $sum: 1 }
43   } }
44 ])
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
```

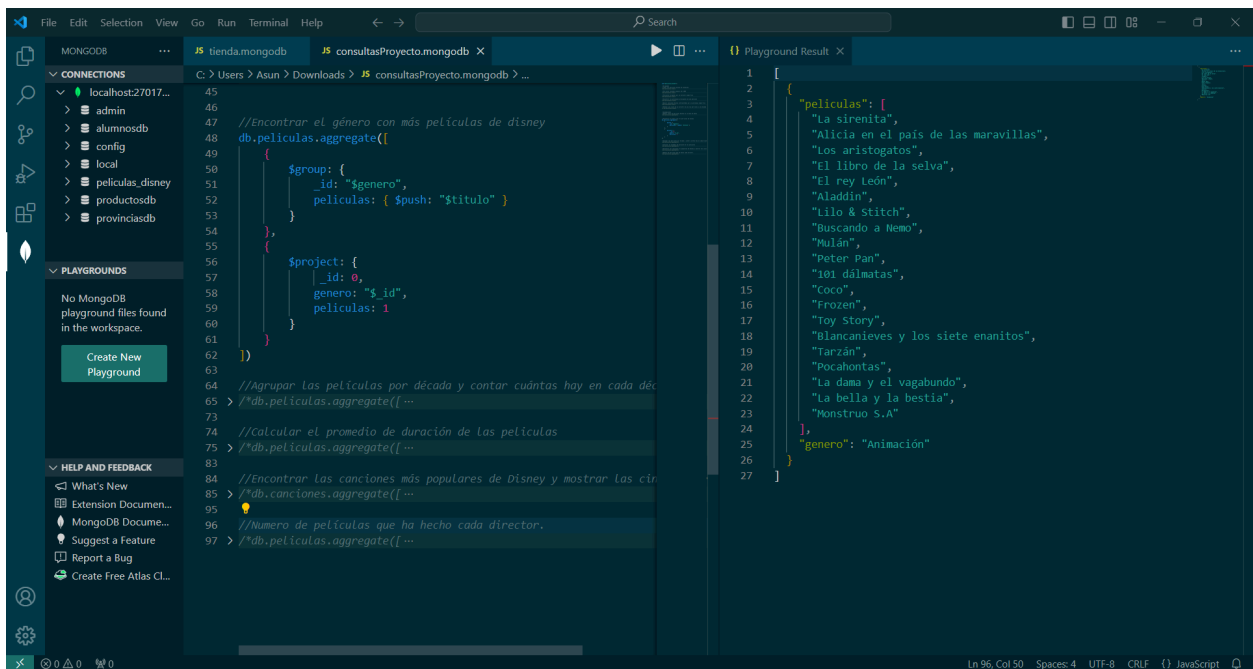
```
1 [
2   {
3     "_id": {
4       "$oid": "6649e8f37e1ff1f58205bbee"
5     },
6     "titulo": "Buscando a Nemo",
7     "añoLanzamiento": 2003,
8     "década": "Década de los 2000",
9     "genero": "Animación",
10    "duracion": 101,
11    "director": [
12      "Andrew Stanton",
13      "Lee Unkrich"
14    ],
15    "descripcion": "El pececillo Nemo, que es hijo único, es muy qu",
16    "valoracion": 4
17  },
18  {
19    "_id": {
20      "$oid": "6649e8f37e1ff1f58205bef2"
21    },
22    "titulo": "Coco",
23    "añoLanzamiento": 2017,
24    "década": "Década de los 2000",
25    "genero": "Animación",
26    "duracion": 109,
27    "director": [
28      "Lee Unkrich",
29      "Adrián Molina"
30    ],
31    "descripcion": "Miguel es un joven con el sueño de convertirse",
32    "valoracion": 4
33  },
34  {
35    "_id": {
36      "$oid": "6649e8f37e1ff1f58205befa"
37    },
38    "titulo": "Monstruo S.A.",
39  }
40 ]
```

5. Agregaciones

- Calcular el número total de películas de Disney en la base de datos.



- Encontrar el género con más películas de Disney.



- Agrupar las películas por década y contar cuántas hay en cada década.

```
39
40 //Agregaciones
41 //Numero total de películas Disney en la base de datos
42 > /*db.películas.aggregate(...)
43
44
45
46
47 //Encontrar el género con más películas de Disney
48 > /*db.películas.aggregate(...)
49
50
51
52
53
54 //Agrupar las películas por década y contar cuántas hay en cada década.
55 db.películas.aggregate([
56   {
57     $group: {
58       _id: "$decada",
59       totalPelículas: { $sum: 1 }
60     }
61   }
62 ])
63
64
65
66
67
68
69
70
71
72
73
74 //Calcular el promedio de duración de las películas
75 > /*db.películas.aggregate(...)
76
77
78
79 //Encontrar las canciones más populares de Disney y mostrar las cinco primeras
80 > /*db.canciones.aggregate(...)
81
82
83
84
85
86 //Numero de películas que ha hecho cada director.
87 > /*db.películas.aggregate(...
```

```
1
2
3 {
4   "_id": "Década de los 80",
5   "totalPelículas": 1
6 },
7 {
8   "_id": "Década de los 50",
9   "totalPelículas": 3
10 },
11 {
12   "_id": "Década de los 60",
13   "totalPelículas": 2
14 },
15 {
16   "_id": "Década de los 40",
17   "totalPelículas": 1
18 },
19 {
20   "_id": "Década de los 70",
21   "totalPelículas": 1
22 },
23 {
24   "_id": "Década de los 90",
25   "totalPelículas": 7
26 },
27 {
28   "_id": "Década de los 2000",
29   "totalPelículas": 5
30 }
```

- Calcular el promedio de duración de las películas de Disney.

```
48 > /*db.películas.aggregate(...)
49
50
51
52
53
54 //Agrupar las películas por década y contar cuántas hay en cada década.
55 db.películas.aggregate([
56   {
57     $group: {
58       _id: "$decada",
59       totalPelículas: { $sum: 1 }
60     }
61   }
62 ])
63
64
65
66
67
68
69
70
71
72
73
74 //Calcular el promedio de duración de las películas
75 db.películas.aggregate([
76   {
77     $group: {
78       "_id": null,
79       duracionPromedioPelicula: { "$avg": "$duracion" }
80     }
81   }
82 ])
83
84
85
86 //Encontrar las canciones más populares de Disney y mostrar las cinco primeras
87 > /*db.canciones.aggregate(...)
88
89
90
91
92
93
94
95
96 //Numero de películas que ha hecho cada director.
97 > /*db.películas.aggregate(...
```

```
1
2
3 {
4   "_id": null,
5   "duracionPromedioPelicula": 84.7
6 }
```

- Encontrar las canciones más populares de Disney y mostrar las cinco primeras.

The screenshot shows the VS Code interface with the MongoDB Playground extension. The left sidebar displays the 'CONNECTIONS' and 'PLAYGROUNDS' panels. The main editor shows a JavaScript file named 'consultasProyecto.mongodb.js' with the following code:

```
48 > //db.peliculas.aggregate({ ...
63
64 //Agrupar las películas por década y contar cuántas hay en cada década
65 > /*db.peliculas.aggregate({ ...
73
74 //Calcular el promedio de duración de las películas
75 > /*db.peliculas.aggregate({ ...
83
84 //Encontrar las canciones más populares de Disney y mostrar las cinco
85 db.canciones.aggregate({
86 {
87   $sort: {
88     "reproducciones": -1
89   },
90 },
91 {
92   $limit: 5
93 }
94 })
95
96 //Numero de películas que ha hecho cada director.
97 > /*db.peliculas.aggregate({ ...
```

The 'PLAYGROUND RESULT' panel on the right shows the output of the aggregation pipeline, displaying a list of movie objects with fields like 'id', 'soid', 'titulo', 'película', 'personajes', and 'reproducciones'.

- Añade una agregación personalizada.

The screenshot shows the VS Code interface with the MongoDB Playground extension. The left sidebar displays the 'CONNECTIONS' and 'PLAYGROUNDS' panels. The main editor shows a JavaScript file named 'consultasProyecto.mongodb.js' with the following code:

```
48 > //db.peliculas.aggregate({ ...
63
64 //Agrupar las películas por década y contar cuántas hay en cada década
65 > /*db.peliculas.aggregate({ ...
73
74 //Calcular el promedio de duración de las películas
75 > /*db.peliculas.aggregate({ ...
83
84 //Encontrar las canciones más populares de Disney y mostrar las cinco
85 db.canciones.aggregate({ ...
95
96 //Numero de películas que ha hecho cada director.
97 db.peliculas.aggregate({
98 {
99   $unwind: "$director"
100 },
101 {
102   $group: {
103     "id": "$director",
104     "peliculasDirector": { $sum: 1 }
105   }
106 }
107 })
```

The 'PLAYGROUND RESULT' panel on the right shows the output of the aggregation pipeline, displaying a list of movie objects with fields like 'id', 'peliculasDirector', and 'reproducciones'.

6. Conexión con Python

Se ha creado en un archivo .py llamado conexionPython en el cual hemos desarrollado toda la conexión entre la base de datos de Mongo y Pycharm, a parte, de un menú de una aplicación con el que se da desarrollado funcionalidades de una aplicación.

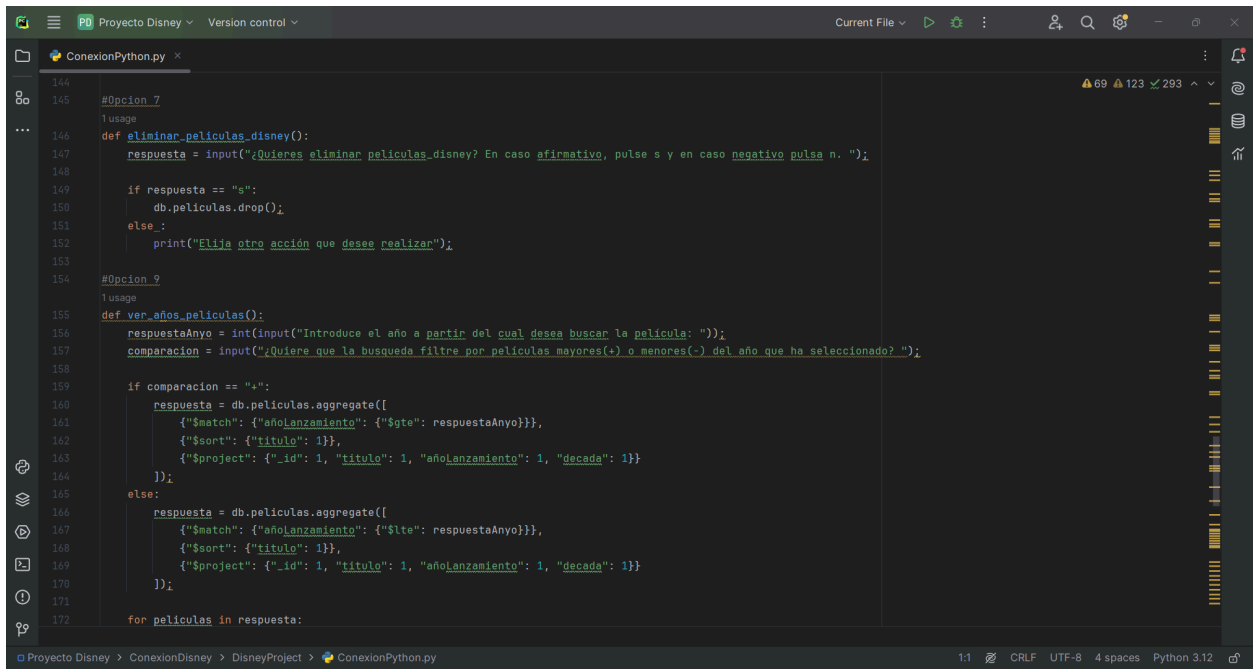
Al principio del archivo.py se han definido unas funciones que se corresponden con las funcionalidades que va hacer la aplicación llamada Disney Cinema. Tenemos nueve funciones: registrar películas nuevas en el catálogo, actualizar la valoración de una película que hayas visto, eliminar películas, ver todas las películas del catálogo, buscar por ..., eliminar todas las películas que haya registradas, eliminar peliculas_disney, salir y ver los años de las películas.


```
Proyecto Disney  Version control  Current File  69 123 293
ConexionPython.py
41
42 #opcion 2
43 def actualizar_valoracion_peliculas():
44     opcion = input("¿Cuántas películas desea actualizar?, si quiere actualizar una pulse 1, si quiere actualizar varias pulse 2: ");
45     if opcion == "1":
46         peliculas = input("¿A qué película desea cambiarle la valoración? ");
47         nuevaValoracion = int(input("Introduce la nueva valoración que tendrá la película: "));
48
49         db.peliculas.update_one(
50             {"titulo": peliculas},
51             {"$set": {"valoracion": nuevaValoracion}}
52         )
53         print("Se ha actualizado la valoración de la película " + peliculas)
54     else:
55         opcion1 = input("¿Quieres buscar por título(t) o por director(d)? ");
56         if opcion1 == "t":
57             titulo = input("Introduce el título de la película que quieres actualizar: ");
58             nuevaValoracion = input("Introduce la nueva valoración que tendrá la película: ");
59
60             db.peliculas.update_many(
61                 {"titulo": titulo},
62                 {"$set": {"valoracion": nuevaValoracion}}
63             )
64         else:
65             director = input("Introduce el nombre del director que quieres actualizar: ");
66             valoracion = input("Introduce la nueva valoración que tendrá la película: ");
67
68             db.libros.update_many(
69                 {"director": director},
70                 {"$set": {"valoracion": valoracion}}
```

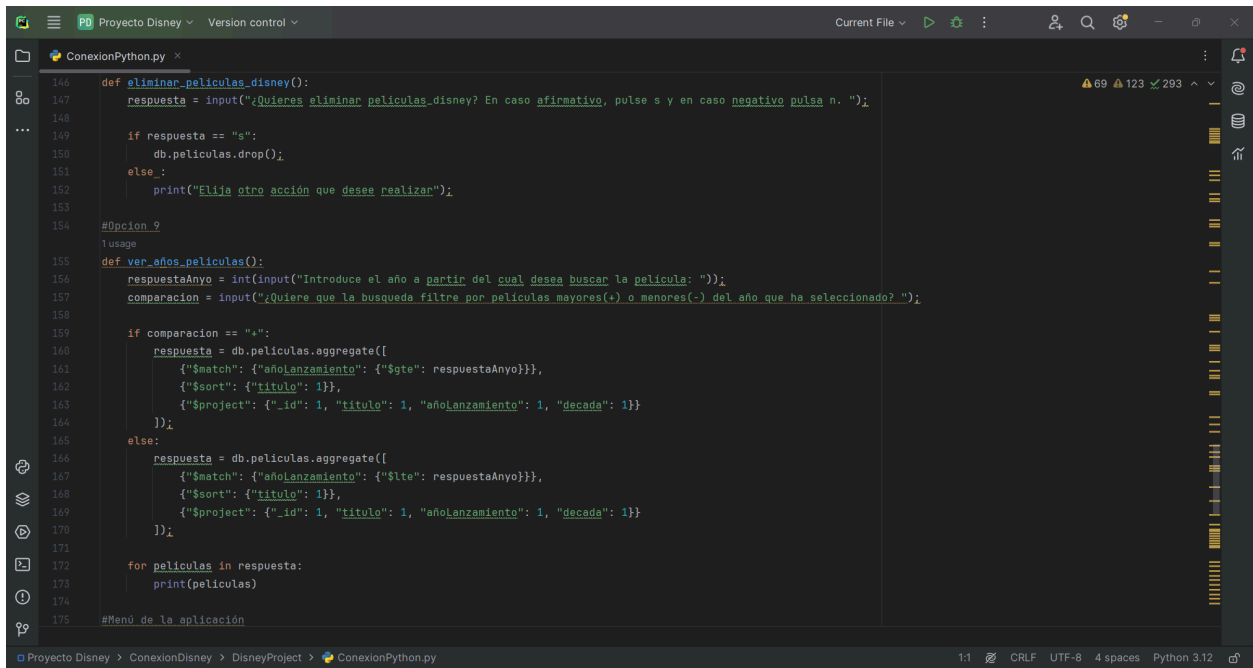
```
Proyecto Disney  Version control  Current File  69 123 293
ConexionPython.py
70         {"$set": {"valoracion": valoracion}}
71     )
72
73 #opcion 3
74 def eliminar_solo_una_pelicula():
75     peliculaEliminar = input("Introduce el título de la película que deseas eliminar: ")
76     respuestaUs = input("¿Quieres eliminar la película " + peliculaEliminar + "? En caso afirmativo, pulse s y en caso negativo pulse n. ");
77     if respuestaUs == "s":
78         db.peliculas.delete_one(
79             {"titulo": peliculaEliminar}
80         )
81         print("Se ha eliminado la película " + peliculaEliminar);
82
83
84 #opcion 4
85 def ver_peliculas_disponibles():
86     peliculasDisponibles = db.peliculas.find()
87     for peliculas in peliculasDisponibles:
88         print(peliculas)
89
```

```
89
90
91 #opcion 5
92 def buscar_peliculas():
93     print("Selecciona una de las siguientes opciones de búsqueda: ");
94     print("1. Búsqueda por título o director");
95     print("2. Búsqueda por género y número de valoración");
96     print("3. Búsqueda por título y década")
97     opcionBuscar = input("Marque con un número del 1 al 3 la opción que desea: ")
98
99     if opcionBuscar == "1":
100         tituloDirector = input("Introduce el título o el director de la película que deseas buscar: ");
101
102         busqueda = db.peliculas.find(
103             {"$or": [{"titulo": tituloDirector}, {"director": tituloDirector}]}
104         )
105     elif opcionBuscar == "2":
106         genero = input("Introduce el género de la película que deseas buscar: ");
107         numeroValoracion = int(input("Introduce el número de valoración por el que deseas filtrar: "));
108         comparacion = input("¿Quiere que la búsqueda filtre por películas mayores(+) o menores(-) de la valoración insertada?: ");
109
110         if comparacion == "+":
111             busqueda = db.peliculas.find(
112                 {"$and": [{"genero": genero}, {"valoracion": {"$gte": numeroValoracion}}]}
113             );
114         else:
115             busqueda = db.peliculas.find(
116                 {"$and": [{"genero": genero}, {"valoracion": {"$lte": numeroValoracion}}]}
117             );
118     elif opcionBuscar == "3":
```

```
118     elif opcionBuscar == "3":
119         titulo = input("Introduce el título de la película que deseas buscar: ");
120         decada = input("Introduce la década de la película que deseas buscar: ");
121
122         busqueda = db.peliculas.find(
123             {"$and": [{"titulo": titulo}, {"decada": decada}]}
124         )
125     else:
126         busqueda = [];
127         print("Elige bien");
128
129     if opcionBuscar >= "1" and opcionBuscar <= "3":
130         for peliculas in busqueda:
131             print(peliculas)
132
133
134 #opcion 6
135 def eliminar_peliculas():
136     respuestaUs = input("¿Quieres eliminar todas las películas? En caso afirmativo, pulse s y en caso negativo pulsa n.");
137
138     if respuestaUs == "s":
139         db.peliculas.delete_many({});
140         print("Se han borrado las películas")
141     else:
142         print("Elija otro acción que desee realizar");
143
144 #opcion 7
145 def eliminar_peliculas_disney():
```



```
144
145 #opcion 7
146 def eliminar_peliculas_disney():
147     respuesta = input("¿Quieres eliminar peliculas_disney? En caso afirmativo, pulse s y en caso negativo pulsa n. ")
148
149     if respuesta == "s":
150         db.peliculas.drop()
151     else:
152         print("Elija otro acción que desee realizar")
153
154 #opcion 9
155 #usage
156 def ver_años_peliculas():
157     respuestaAnyo = int(input("Introduce el año a partir del cual desee buscar la película: "))
158     comparacion = input("¿Quiere que la búsqueda filtre por películas mayores(+) o menores(-) del año que ha seleccionado? ")
159
160     if comparacion == "+":
161         respuesta = db.peliculas.aggregate([
162             {"$match": {"añoLanzamiento": {"$gte": respuestaAnyo}}},
163             {"$sort": {"titulo": 1}},
164             {"$project": {"_id": 1, "titulo": 1, "añoLanzamiento": 1, "decada": 1}}
165         ])
166     else:
167         respuesta = db.peliculas.aggregate([
168             {"$match": {"añoLanzamiento": {"$lte": respuestaAnyo}}},
169             {"$sort": {"titulo": 1}},
170             {"$project": {"_id": 1, "titulo": 1, "añoLanzamiento": 1, "decada": 1}}
171         ])
172     for peliculas in respuesta:
173         print(peliculas)
```



```
146 def eliminar_peliculas_disney():
147     respuesta = input("¿Quieres eliminar peliculas_disney? En caso afirmativo, pulse s y en caso negativo pulsa n. ")
148
149     if respuesta == "s":
150         db.peliculas.drop()
151     else:
152         print("Elija otro acción que desee realizar")
153
154 #opcion 9
155 #usage
156 def ver_años_peliculas():
157     respuestaAnyo = int(input("Introduce el año a partir del cual desee buscar la película: "))
158     comparacion = input("¿Quiere que la búsqueda filtre por películas mayores(+) o menores(-) del año que ha seleccionado? ")
159
160     if comparacion == "+":
161         respuesta = db.peliculas.aggregate([
162             {"$match": {"añoLanzamiento": {"$gte": respuestaAnyo}}},
163             {"$sort": {"titulo": 1}},
164             {"$project": {"_id": 1, "titulo": 1, "añoLanzamiento": 1, "decada": 1}}
165         ])
166     else:
167         respuesta = db.peliculas.aggregate([
168             {"$match": {"añoLanzamiento": {"$lte": respuestaAnyo}}},
169             {"$sort": {"titulo": 1}},
170             {"$project": {"_id": 1, "titulo": 1, "añoLanzamiento": 1, "decada": 1}}
171         ])
172     for peliculas in respuesta:
173         print(peliculas)
174
175 #Menú de la aplicación
```

Tras desarrollar cada una de las funcionalidades de la aplicación, se ha creado un menú en el que se realizan las llamadas a las funciones que hemos diseñado previamente

```
175 #Menú de la aplicación
176 opcion = 0;
177 print("Bienvenido a la aplicación Disney Cinema.")
178 while opcion != "8":
179
180     print("¿qué es lo que desea hacer?");
181
182     print("1: Registrar una o varias películas nuevas en el catálogo");
183     print("2: Actualizar la valoración de una película que hayas visto");
184     print("3: Eliminar una película");
185     print("4: Ver todas las películas del catálogo");
186     print("5: Buscar por...");
187     print("6: Eliminar todos las películas que haya registradas");
188     print("7: Eliminar películas disney");
189     print("8: Salir");
190     print("9: Ver los años de las películas");
191
```

```
193 opcion = input("Marque con un número del 1 al 9 la opción que desea: ")
194
195 if opcion == "1":
196     registrar_peliculas();
197 elif opcion == "2":
198     actualizar_valoracion_peliculas();
199 elif opcion == "3":
200     eliminar_solo_una_pelicula();
201 elif opcion == "4":
202     ver_peliculas_disponibles();
203 elif opcion == "5":
204     buscar_peliculas();
205 elif opcion == "6":
206     eliminar_peliculas();
207 elif opcion == "7":
208     eliminar_peliculas_disney();
209 elif opcion == "8":
210     print("Salir");
211 elif opcion == "9":
212     ver_años_peliculas();
213 else:
214     print("Vuelve a introducir un valor válido entre 1 y 8");
215
216
217
218
ver_años_peliculas()
```

■ ■ ■