

Universitat Autònoma de Barcelona

MATEMÀTIQUES COMPUTACIONALS I ANÀLISI DE DADES

PROJECTE DE MONGODB

Bases de dades no relacionals

Autors:

Alberto Real: 1544112, Judit Panisello: 1605512

Biel González: 1551813, Cristina Soler: 1603542

Repositori de GitHub: <https://github.com/kermitsc7/Projecte-MongoDB>

Abril 2022

Contents

1	Introducció	2
2	Objectius	2
3	Repartició de tasques	2
4	Exercici 1. Patrons de Disseny	3
4.1	Col·lecció “Col·lecció” en el JSON	3
4.2	Col·lecció “Editorial” en el JSON	4
4.3	Col·lecció “Personatge” en el JSON	4
4.4	Col·lecció “Artista” en el JSON	5
4.5	Col·lecció “Publicació” en el JSON	5
5	Exercici 2. Script de Python	6
5.1	Script de Python:	7
6	Exercici 3. Joc de Proves	9
6.1	Exercici 1	9
6.2	Exercici 2	9
6.3	Exercici 3	10
6.4	Exercici 4	10
6.5	Exercici 5	11
6.6	Exercici 6	11
6.7	Exercici 7	12
6.8	Exercici 8	12
6.9	Exercici 9	13
6.10	Exercici 10	14

1 Introducció

Tenim una botiga de còmics i necessitem fer inventari per així facilitar el nostre treball dins de la botiga. A l'hora de guardar la informació de cada publicació també volem guardar quins artistes hi han participat, els personatges que surten a cada publicació i a quina col·lecció pertanyen, a més a més de l'editorial que els ha editat.

Per dur a terme aquesta tasca farem servir una base de dades no relacional documental, ja que als del departament tecnològic de la botiga ens sembla el més adequat i més s'adapta a les nostres necessitats. Un cop tinguem feta la base de dades voldrem comprovar que aquesta funciona correctament i ens permet buscar la informació que volem.

2 Objectius

Els objectius d'aquest treball són:

- Extreure a partir d'un disseny entitat-relació un conjunt de col·leccions connectades entre elles amb patrons de disseny segons les relacions entre cadascuna de les entitats que tenim i les seves respectives relacions.
- Implementar un script de Python que ens permeti passar la informació que tenim a un arxiu “.xlsx” amb els dissenys de cada col·lecció que hem fet anteriorment i amb els seus tipus corresponents.
- Realitzar una sèrie de consultes des del NoSQLBooster a la nostra base de dades per veure si funciona correctament i demostrar el nostre coneixement sobre les consultes amb MongoDB.
- Crear i mantenir actualitzat un GitHub per guardar el nostre progrés amb el projecte i així tenir accés global al nostre avenç i visualitzar els canvis al nostre treball.

3 Repartició de tasques

- **Alberto:** JSON i queries (4,7,10)
- **Cris:** Model Entitat-Relacio i queries (3,6,9)
- **Judit:** Script de Python i queries (1)
- **Biel:** Correcció d'errors (codi i ortografia) i queries (2,5,8)

L'informe ha estat fet entre tots. Encara que la repartició ha estat l'anterior, s'han resolt els dubtes i problemes en conjunt.

4 Exercici 1. Patrons de Disseny

Se'ns ha proporcionat el següent model Entitat - Relació:

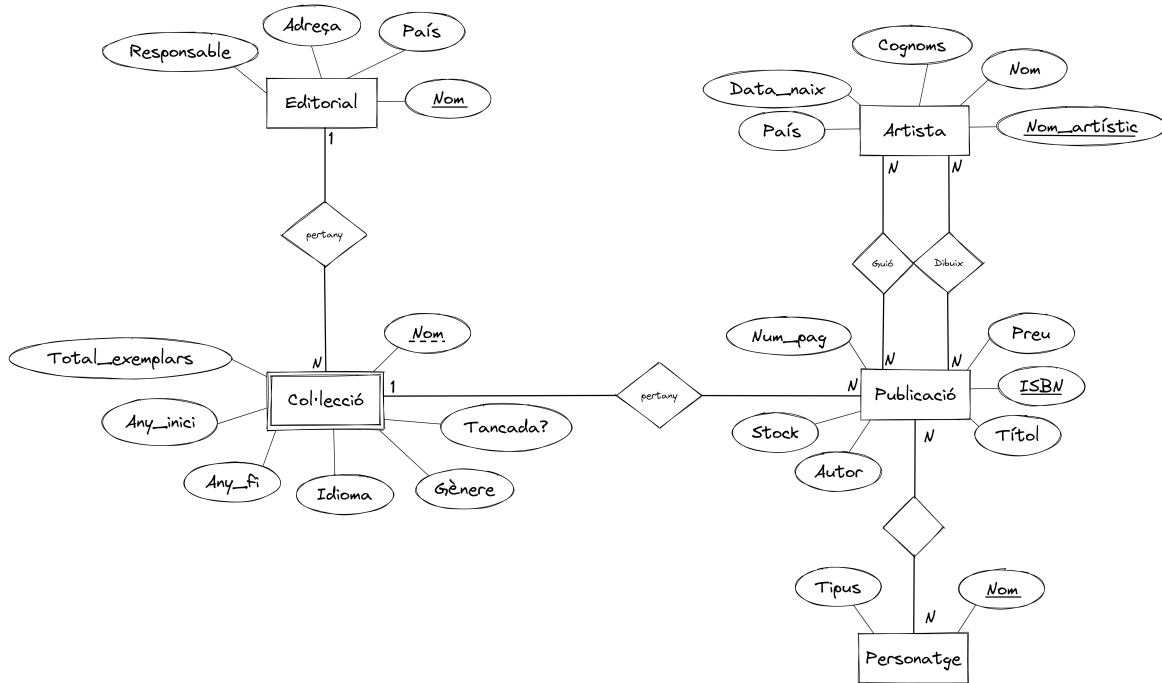


Figure 1: Model relacional de la base de dades

En aquest model Entitat-Relació tenim quatre entitats, tres entitats fortes i una dèbil, les quals són “Editorial”, “Artista”, “Publicació” i “Col·lecció”, amb els seus respectius atributs, sent “Col·lecció” l’entitat dèbil.

A partir del model Entitat-Relació construïrem l’estructura JSON amb la qual realitzarem les consultes.

4.1 Col·lecció “Col·lecció” en el JSON

```
Col·lecció <collection>{
  "nom": <string>
  "total_exemplars": <int>
  "genere": [<string>, <string>]
  "idioma": <string>
  "any_inici": <int>
  "any_fi": <int>
  "tancada": <boolean>
  "nom_editorial": <string>
}
```

Tipus de patró: Referencial

Figure 2: JSON “col·leccions”

A la col·lecció “Col·leccions” tenim els atributs corresponents a aquesta col·lecció amb els tipus de dades adients per a cadascun. A més a més, hem afegit l’atribut “nom_editorial”, que serà de tipus “string”, per

relacionar aquesta col·lecció amb la col·lecció “Editorial”. Aquí hem fet servir un patró referencial per connectar les dues col·leccions, només utilitzarem un atribut de la col·lecció “Editorial” per evitar tenir col·leccions amb molts atributs.

4.2 Col·lecció “Editorial” en el JSON

```
Editorial <collection>{  
  "nom":      <string>  
  "adreça":   <string>  
  "responsable": <string>  
  "pais":     <string>  
}
```

Figure 3: JSON “Editorial”

A la col·lecció “Editorial” tenim els atributs corresponents a aquesta col·lecció amb els tipus de dades adients per a cada un. No fem ús de cap mena de patró, ja que aquesta ja està relacionada amb “Col·lecció” des d’aquesta mateixa.

4.3 Col·lecció “Personatge” en el JSON

```
Personatge <collection>{  
  "nom":      <string>  
  "tipus":    <string>  
  "ISBN":     <int>  
}
```

Tipus de patró: Referencial

Figure 4: JSON “Personatge”

A la col·lecció “Personatge” tenim els atributs corresponents a aquesta col·lecció amb els tipus de dades adients per a cadascun. A part hem afegit l’atribut “ISBN” que serà de tipus “int” per relacionar-la aquesta col·lecció amb la col·lecció “Publicació”, això torna a ser un patró referencial, ja que només utilitzem un atribut de la col·lecció “Publicació” per evitar tenir col·leccions amb els mateixos atributs repetits.

4.4 Col·lecció “Artista” en el JSON

```
Artista <collection>{
  "nom_artistic":    <string>
  "nom":             <string>
  "cognoms":         <int>
  "data_naixement":  <DATE>
  "pais":            <string>
}
```

Figure 5: JSON “Artista”

A la col·lecció “Artista” tenim els atributs corresponents a aquesta col·lecció amb els tipus de dades adients per a cada un. No fem ús de cap mena de patró, ja que aquesta ja està relacionada amb “Publicació” des d’aquesta mateixa, com veurem a continuació.

4.5 Col·lecció “Publicació” en el JSON

```
Publicació <collection>{
  "ISBN":            <int>
  "autor":           <string>
  "preu":            <float>
  "num_pagines":     <int>
  "stock":           <int>
  "titol":           <string>
  "guionistes":      [<string>, <string>]
  "dibuixants":      [<string>, <string>]
  "nom_col·lecció": <string>
  "nom_editorial":   <string>
}
```

Tipus de patró: Referencial

Figure 6: JSON “Publicació”

A la col·lecció “Publicació” tenim els atributs corresponents a aquesta col·lecció amb els tipus de dades adients per a cadascun. A més a més, hem afegit quatre atributs: “nom_col·lecció”, “nom_edició”, “guionistes” i “dibuixants”, els dos primers són per enllaçar amb col·lecció i edició respectivament, mentre que dos últims atributs són llistes que guardaran els “strings” amb el “nom_artistic” dels guionistes o dibuixants, respectivament, dins d’una llista. Això ens servirà per relacionar aquesta col·lecció amb la col·lecció “Artista” tenint en compte si l’artista és guionista o dibuixant, per fer això utilitzem el patró referencial per evitar tenir col·leccions amb els mateixos atributs repetits.

5 Exercici 2. Script de Python

Tal com ens indica l'enunciat, hem de generar un script de Python el qual ens permeti llegir les dades en *.xlsx* i crear les diferents col·leccions prèviament dissenyades i guardar-les com *.json*.

Realitzarem un control d'errors, si l'script s'executa més d'una vegada o s'interromp a la meitat de l'execució, el que implica que una part de la base de dades no és inserida, en tornar a executar l'script el que farem per prevenir possibles duplicacions, és eliminar la base de dades ja creada. Per tant, a l'inici del script mirarem si la base de dades està creada i/o plena eliminarem la base de dades i tornarà a ser inserida al complet.

Si la base de dades fos molt grossa, això podria suposar un problema, ja que estar constantment eliminant i creant la base de dades seria molt costós. Però en el nostre cas hem considerat que la base de dades és petita, per tant, no suposarà molt temps d'execució ni serà costós.

5.1 Script de Python:

```
import pandas as pd
from pymongo import MongoClient

csv_path = 'Dades.xlsx'
db_name = 'Lliurament'

# Leer el archivo XLSX
xl_file = pd.read_excel('Dades.xlsx', sheet_name=None)

# Conectarse a la base de datos MongoDB

client = MongoClient('mongodb://dcccluster.uab.cat:8227/')

#Eliminamos Base de Datos
client.drop_database(db_name)

# Creamos Base de Datos
db = client[db_name]

# Miramos si la coleccion esta creada y si lo esta la eliminamos
noms = ['Colleccions', 'Publicacions', 'Editorial', 'Artistes', 'Personatges']
for col in noms:
    if col in db.list_collection_names():
        db[col].drop()

# Crear una coleccin por cada pestaa
for sheet_name, df in xl_file.items():

    if sheet_name == 'Colleccions-Publicacions':

        collectionC = db['Colleccions']
        collectionP = db['Publicacions']
        collectionE = db['Editorial']
        df = xl_file[sheet_name]

        # Colleccions
        dfC = df[['NomColleccio', 'total_exemplars', 'genere', 'idioma',
                  'any_inici', 'any_fi', 'tancada', 'NomEditorial']]
        dfC.genere = dfC.genere.apply(lambda x: x[1:-1].split(', '))
        dfC=dfC.drop_duplicates(subset=["NomColleccio", "NomEditorial"])
        dataC = dfC.to_dict('records')

        #Publicacions
```



```

dfP = df[['ISBN', 'titol', 'stock', 'autor', 'preu', 'num_pagines',
         'guionistes', 'dibuixants', 'NomColleccio', 'NomEditorial']]
dfP.guionistes = dfP.guionistes.apply(lambda x: x[1:-1].split(' '))
dfP.dibuixants = dfP.dibuixants.apply(lambda x: x[1:-1].split(' '))
dataP = dfP.to_dict('records')

#Editorial
dataE = df[['NomEditorial', 'resposable',
         'adreca', 'pais']].to_dict('records')

collectionC.insert_many(dataC)
collectionP.insert_many(dataP)
collectionE.insert_many(dataE)
else:
    # Obtener el nombre de la coleccion
    collection_name = sheet_name
    # Insertar los datos en la coleccion
    collection = db[collection_name]
    data = df.to_dict('records')
    collection.insert_many(data)

# Contar el numero de colecciones creadas
num_collections = len(db.list_collection_names())

# Imprimir el numero de colecciones creadas
print(f'Se crearon {num_collections} colecciones en la base de datos.')

```

[

6 Exercici 3. Joc de Proves

6.1 Exercici 1

Les 5 publicacions amb major preu. Mostrar només el títol i preu.

```

1 // 1. Les 5 publicacions amb major preu. Mostrar només el títol i preu.
2 db.Publicacions.find().sort({'preu':-1}).limit(5).projection({'titol':1,'preu':1,'_id':0})
3

```

Key	Value
▶ (1)	{ titol : "Dracula", preu : 125.5 }
▶ (2)	{ titol : "Tragedias", preu : 85.4 }
▶ (3)	{ titol : "Romances", preu : 72.4 }
▶ (4)	{ titol : "Crimen y castigo", preu : 59.4 }
▶ (5)	{ titol : "En el Este", preu : 43.5 }

Figure 7: Exercici 1

6.2 Exercici 2

Valor màxim, mínim i mitjà del preus de les publicacions de l'editorial Juniper Books.

```

5 // 2. Valor màxim, mínim i mitjà del preus de les publicacions de l'editorial Juniper Books
6 db.Publicacions.aggregate([
7   {$match: {"NomEditorial":"Juniper Books"}},
8   {$group: {_id:"$NomEditorial", "Max Price":{$max:"$preu"}, "Min Price":{$min:"$preu"}, "Avg Price":{$avg:"$preu"}}},
9 ])
10

```

Key	Value
▶ (1) Juniper Books	{ "Max Price" : 32.5, "Min Price" : 27.85, "Avg Price" : 29.118181818182 } (4 fields)
_id	Juniper Books
Max Price	32.5
Min Price	27.85
Avg Price	29.1182

Figure 8: Exercici 2

6.3 Exercici 3

Artistes (nom artístic) que participen en més de 5 publicacions com a dibuixant.

```
12 // 3. Artistes (nom artístic) que participen en més de 5 publicacions com a dibuixant.
13 db.Publicacions.aggregate([
14   {$unwind: "$dibuixants"},
15   {$sortByCount: "$dibuixants"},
16   {$match: {"count": {$gt: 5}}},
17   {$project: {"dibuixants": 1}}
18 ])
```

Find x Aggregate x Aggregate (1) x

Publicacions 0.103 s 2 Docs

Key	Value
(1) Artista1	{}
_id	Artista1
(2) Artista2	{}
_id	Artista2

Figure 9: Exercici 3

6.4 Exercici 4

Numero de col·leccions per gènere. Mostra gènere i número total.

```
21 // 4. Numero de col·leccions per gènere. Mostra gènere i número total
22 db.Colleccions.aggregate([
23   {$unwind: '$genere'},
24   {$sortByCount: '$genere'}
25 ])
```

Find x Aggregate x Aggregate (1) x Aggregate (2) x

Colleccions 0.173 s 5 Docs

Key	Value
(1) fantasia	{ count : 4 }
(2) belica	{ count : 2 }
(3) magia	{ count : 2 }
(4) suspense	{ count : 1 }
(5) clasicos	{ count : 1 }

Figure 10: Exercici 4

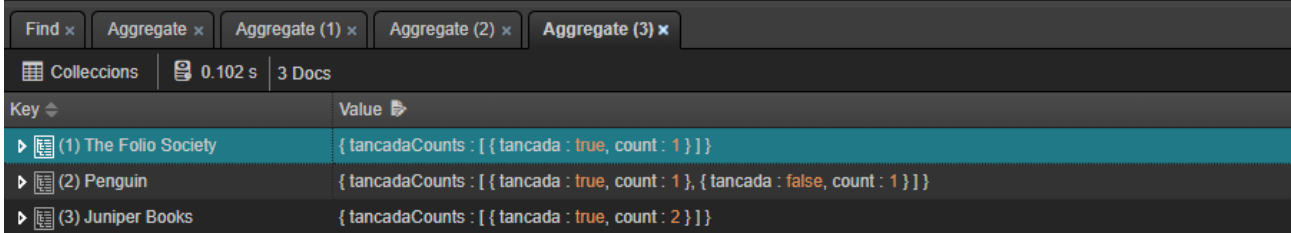
6.5 Exercici 5

Per cada editorial, mostrar el recompte de col·leccions finalitzades i no finalitzades.

```

28 // 5. Per cada editorial, mostrar el recompte de col·leccions finalitzades i no finalitzades.
29 db.Colleccions.aggregate([
30   {$group: {_id: {NomEditorial: "$NomEditorial", tancada: "$tancada"}, count: {$sum: 1}}},
31   {$group: {_id: "$_id.NomEditorial", tancadaCounts: {$push: {tancada: "$_id.tancada", count: "$count"}}}}
32 ])
33

```



Key	Value
(1) The Folio Society	{ tancadaCounts : [{ tancada : true, count : 1 }]}
(2) Penguin	{ tancadaCounts : [{ tancada : true, count : 1 }, { tancada : false, count : 1 }]}
(3) Juniper Books	{ tancadaCounts : [{ tancada : true, count : 2 }]}

Figure 11: Exercici 5

En aquest exercici, veiem que els “tancada:true” són aquelles col·leccions finalitzades, mentre que “tancada:false” són aquelles que no estan finalitzades.

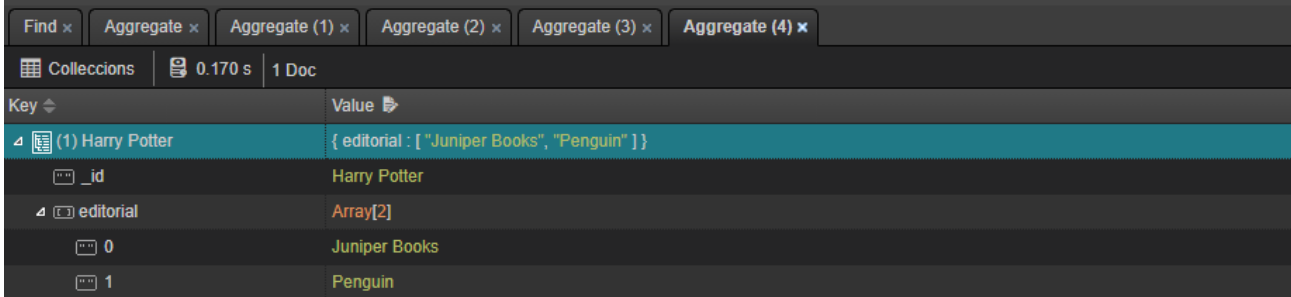
6.6 Exercici 6

Mostrar les 2 col·leccions ja finalitzades amb més publicacions. Mostrar editorial i nom col·lecció.

```

35 // 6. Mostrar les 2 col·leccions ja finalitzades amb més publicacions. Mostrar editorial i nom col·lecció.
36 db.Colleccions.aggregate([
37   {$match: {tancada: true}},
38   {$group: {_id: '$NomColleccio', 'editorial': {$addToSet: '$NomEditorial'}}},
39   {$match: {'editorial.1': {$exists: true}}}
40 ])
41

```



Key	Value
(1) Harry Potter	{ editorial : ["Juniper Books", "Penguin"] }

Figure 12: Exercici 6

6.7 Exercici 7

Mostrar el país d'origen de l'artista o artistes que han fet més guions.

```

43 // 7. Mostrar el país d'origen de l'artista o artistes que han fet més guions.
44 db.Publicacions.aggregate([
45   {$unwind:'$guionistes'},
46   {$sortByCount:'$guionistes'},
47   {$lookup:{ from: "Artistes", localField: "_id", foreignField: "Nom_artistic", as: "match"}},
48   {$project:{ "match.pais":1, '_id':0}},
49   {$limit:1}
50 ])

```

Key	Value
(1)	{ match : [{ pais : "Noruega" }]}
match	Array[1]
0	{ pais : "Noruega" }

Figure 13: Exercici 7

6.8 Exercici 8

Mostrar les publicacions amb tots els personatges de tipus “heroe”.

```

53 // 8. Mostrar les publicacions amb tots els personatges de tipus “heroe”.
54 db.Publicacions.aggregate([
55   {$lookup: {from: "Personatges", localField: "ISBN", foreignField: "isbn", as: "personatges"}},
56   {$match: {"personatges.tipus": "heroe"}},
57   {$group: {_id: "$ISBN", tipus: {$addToSet: "$personatges.tipus"}},
58   {$unwind: "$tipus"},
59   {$project: { _id: "$_id", totsHeroe: {$allElementsTrue: {$map: {input: "$tipus", as: "elem", in: {$eq: ["$elem", "heroe"]}}}}}},
60   {$match: {totsHeroe: true}},
61   {$project: {'_id': 1}}
62 ])

```

Key	Value
(1) 4	{}
_id	4
(2) 20	{}
_id	20
(3) 22	{}
_id	22

Figure 14: Exercici 8

Hem hagut de canviar el fitxer “xlsx”, ja que en aquest exercici hem vist un error tipogràfic on al personatge “Arya Stark” tenia el tipus “heror” en comptes de “heroe”.

6.9 Exercici 9

Modificar el preu de les publicacions amb stock superior a 20 exemplars i incrementar-lo un 25

```

65 // 9. Modificar el preu de les publicacions amb stock superior a 20 exemplars i incrementar-lo un 25%.
66 db.Publicacions.aggregate([
67   {$match: {'stock': {$gt: 20}}},
68   {$project: {"preu": {$multiply: ['$preu', 1.25]}}},
69   {$merge: {into: "Publicacions", on: "_id", whenMatched: "merge", whenNotMatched: 'insert' }}]
70

```

Find x Aggregate x Aggregate (1) x Aggregate (2) x Aggregate (3) x Aggregate (4) x Aggregate (5) x Aggregate (6) x Aggregate (7)

Publicacions 0.225 s 26 Docs

Key	Value
▶ (1)	{ ISBN : 1, stock : 20, preu : 32.5 }
▶ (2)	{ ISBN : 2, stock : 5, preu : 32.5 }
▶ (3)	{ ISBN : 3, stock : 50, preu : 40.625 }
▶ (4)	{ ISBN : 4, stock : 7, preu : 27.85 }
▶ (5)	{ ISBN : 5, stock : 6, preu : 27.85 }
▶ (6)	{ ISBN : 6, stock : 2, preu : 27.85 }
▶ (7)	{ ISBN : 7, stock : 22, preu : 34.8125 }
▶ (8)	{ ISBN : 8, stock : 13, preu : 27.85 }
▶ (9)	{ ISBN : 9, stock : 44, preu : 34.8125 }
▶ (10)	{ ISBN : 10, stock : 7, preu : 27.85 }
▶ (11)	{ ISBN : 11, stock : 9, preu : 27.85 }
▶ (12)	{ ISBN : 12, stock : 15, preu : 31.05 }
▶ (13)	{ ISBN : 13, stock : 60, preu : 38.8125 }
▶ (14)	{ ISBN : 14, stock : 4, preu : 31.05 }
▶ (15)	{ ISBN : 15, stock : 80, preu : 38.8125 }
▶ (16)	{ ISBN : 16, stock : 20, preu : 31.05 }
▶ (17)	{ ISBN : 17, stock : 11, preu : 31.05 }
▶ (18)	{ ISBN : 18, stock : 13, preu : 31.05 }
▶ (19)	{ ISBN : 19, stock : 20, preu : 31.05 }
▶ (20)	{ ISBN : 20, stock : 22, preu : 51.25 }
▶ (21)	{ ISBN : 21, stock : 21, preu : 53.125 }
▶ (22)	{ ISBN : 22, stock : 4, preu : 43.5 }

Figure 15: Exercici 9

6.10 Exercici 10

Mostrar ISBN i títol de les publicacions conjuntament amb tota la seva informació dels personatges.

```

79 // 10. Mostrar ISBN i títol de les publicacions conjuntament amb tota la seva informació dels personatges.
80 db.Publicacions.aggregate([
81   {$lookup:{ from: "Personatges", localField: "ISBN", foreignField: "isbn", as: "personatges"}},
82   {$project:{ "ISBN":1, "títol":1, "personatges":1, "_id":0}}
83 ])

```

Find x Aggregate x Aggregate (1) x Aggregate (2) x Aggregate (3) x Aggregate (4) x Aggregate (5) x Aggregate (6) x Aggregate (7) x

Publicacions 0.151 s 26 Docs

Key	Value
▲ (1)	{ ISBN : 1, títol : "The fellowship of the ring" } (3 fields)
ISBN	1
títol	The fellowship of the ring
▲ personatges	Array[3]
▲ 0	{ _id : ObjectId("642f11f44349de5279e4245f"), nom : "Gandalf", tipus : "mago", isbn : 1 } (4 fields)
_id	642f11f44349de5279e4245f
nom	Gandalf
tipus	mago
isbn	1
▲ 1	{ _id : ObjectId("642f11f44349de5279e42460"), nom : "Frodo", tipus : "heroe", isbn : 1 } (4 fields)
_id	642f11f44349de5279e42460
nom	Frodo
tipus	heroe
isbn	1
▲ 2	{ _id : ObjectId("642f11f44349de5279e42461"), nom : "Samsagaz", tipus : "segundo", isbn : 1 } (4 fields)
_id	642f11f44349de5279e42461
nom	Samsagaz
tipus	segundo
isbn	1
▲ (2)	{ ISBN : 2, títol : "The two towers" } (3 fields)
ISBN	2
títol	The two towers
▶ personatges	Array[4]
▲ (3)	{ ISBN : 3, títol : "The return of the King" } (3 fields)
ISBN	3
títol	The return of the King
▶ personatges	Array[4]
▲ (4)	{ ISBN : 4, títol : "Harry potter y la piedra filosofal" } (3 fields)
ISBN	4

Figure 16: Exercici 10