

1. Create a database called Meflix and collection called Film
2. Insert the following data (the following format is in standard JSON, you might need to modified a bit to suit with Compass, you can see from the demo video). You can use insert one line by line or insert many using array.



Create Database

Database Name

Meflix

Collection Name

Film

☐ Capped Collection

Fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. ⓘ

☐ Use Custom Collation

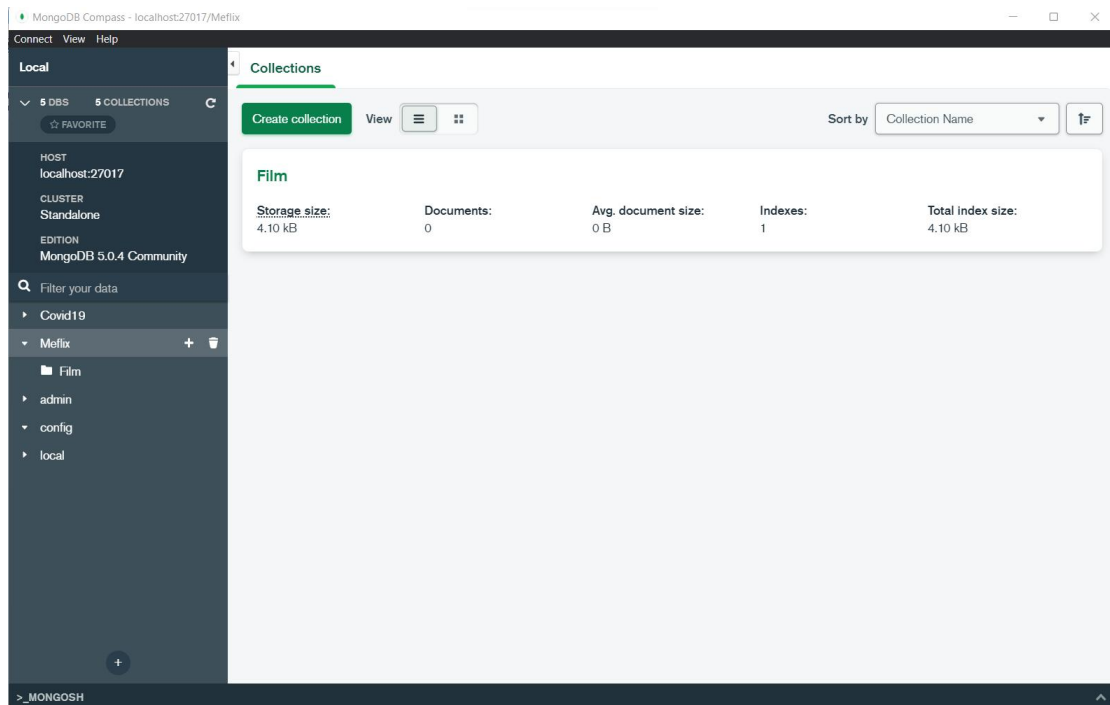
Collation allows users to specify language-specific rules for string comparison, such as rules for lettercase and accent marks. ⓘ

☐ Time-Series

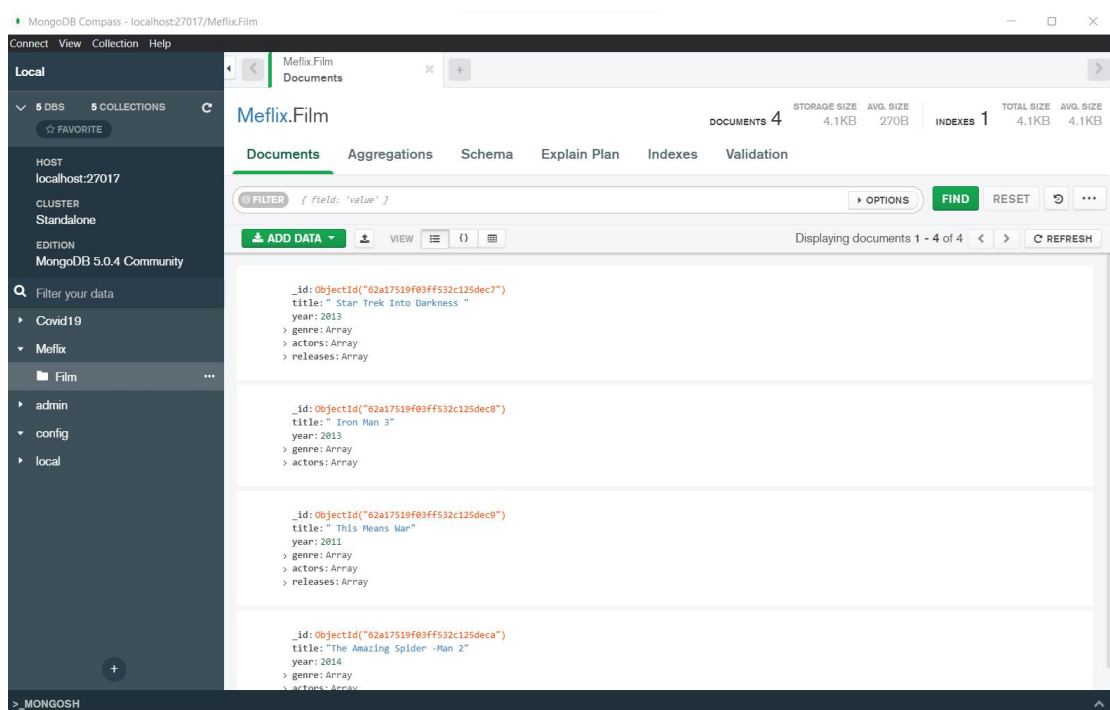
Time-series collections efficiently store sequences of measurements over a period of time.

Cancel

Create Database



```
[ { "title" : " Star Trek Into Darkness ", "year" : 2013 , "genre" : [ " Action " , " Adventure " , "Sci-Fi" ] ,
"actors" : [ "Pine , Chris " , "Quinto , Zachary " , " Saldana , Zoe" ] , "releases" : [ { "country" : "USA" ,
"date" : " 2013 -05 -17 " , "prerelease" : true } , { "country" : " Germany " , "date" : " 2003 -05 -16 " ,
"prerelease" : false } ] } ,
{ "title" : " Iron Man 3" , "year" : 2013 , "genre" : [ " Action " , " Adventure " , "Sci-Fi" ] , "actors" : [ "
Downey Jr. , Robert " , " Paltrow , Gwyneth " ] } ,
{ "title" : " This Means War" , "year" : 2011 , "genre" : [ " Action " , " Comedy " , " Romance " ] , "actors" :
[ "Pine , Chris " , " Witherspoon , Reese " , "Hardy , Tom" ] , "releases" : [ { "country" : "USA" , "date" : "
2011 -02 -17 " , "prerelease" : false } , { "country" : "UK" , "date" : " 2011 -03 -01 " , "prerelease" : true } ] } ,
{ "title" : "The Amazing Spider -Man 2" , "year" : 2014 , "genre" : [ " Action " , " Adventure " , " Fantasy " ] ,
"actors" : [ "Stone , Emma " , " Woodley , Shailene " ] } ] }
```



3. Perform Queries (Read) (Find and projection)

- Write a query that return all film with field title , year, genre and sort according to year Hint: Field: 0 (field not return, Field: 1 (field will return) set in the projection
- Write a query that return movie title Iron man 3, Hint: find ({title: "Iron Man 3"})
- Write a query that return movie with year greater than 2010 and less than or equal to 2013 Hint : $\{ \$and: [{ year: \{ \$gt: 2010 \} }, \{ year: \{ \$lte: 2013 \} }] \}$

Meflix.Film

DOCUMENTS 4 STORAGE SIZE 4.1KB AVG. SIZE 270B INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' } **OPTIONS** **FIND** **RESET** **...**

PROJECT { title: 1, year: 1, genre: 1 }

SORT { year: -1 } **MAX TIME MS** 60000

COLLATION { locale: 'simple' } **SKIP** 0 **LIMIT** 0

VIEW **{} {} {} {}** Displaying documents 1 - 4 of 4 **REFRESH**

```
{
  "_id": ObjectId("62a17519f03ff532c125deca"),
  "title": "The Amazing Spider -Man 2",
  "year": 2014,
  "genre": Array
}
```

```
{
  "_id": ObjectId("62a17519f03ff532c125dec7"),
  "title": " Star Trek Into Darkness ",
  "year": 2013,
  "genre": Array
}
```

```
{
  "_id": ObjectId("62a17519f03ff532c125dec8"),
  "title": " Iron Man 3",
  "year": 2013,
  "genre": Array
}
```

```
{
  "_id": ObjectId("62a17519f03ff532c125dec9"),
  "title": " This Means War",
  "year": 2011,
  "genre": Array
}
```

Meflix.Film

DOCUMENTS 4 STORAGE SIZE 4.1KB AVG. SIZE 270B INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { title: " Iron Man 3" } **OPTIONS** **FIND** **RESET** **...**

PROJECT { field: 0 }

SORT { field: -1 } or [{ 'field', -1 }] **MAX TIME MS** 60000

COLLATION { locale: 'simple' } **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **{} {} {} {}** Displaying documents 1 - 1 of 1 **REFRESH**

```
{
  "_id": ObjectId("62a17519f03ff532c125dec8"),
  "title": " Iron Man 3",
  "year": 2013,
  "genre": Array,
  "actors": Array
}
```

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER `{{ $and: [{ year: { $gt: 2010 } }, { year: { $lte: 2013 } }] }}` **OPTIONS** **FIND** **RESET** **↺** **⋮**

PROJECT `{ field: 0 }`

SORT `{ field: -1 } or [['field', -1]]` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **≡** **{ }** **📄** Displaying documents 1 - 3 of 3 **<** **>** **REFRESH**

```

_id: ObjectId("62a17519f03ff532c125dec7")
title: " Star Trek Into Darkness "
year: 2013
> genre: Array
> actors: Array
> releases: Array

_id: ObjectId("62a17519f03ff532c125dec8")
title: " Iron Man 3 "
year: 2013
> genre: Array
> actors: Array

_id: ObjectId("62a17519f03ff532c125dec9")
title: " This Means War "
year: 2011
> genre: Array
> actors: Array
> releases: Array

```

Query array and nested object

iii. Find the movie, genre="Sci-Fi" , {genre:"Sci -Fi"}

FILTER `{genre:"Sci -Fi"}` **OPTIONS** **FIND** **RESET** **↺** **⋮**

PROJECT `{ field: 0 }`

SORT `{year:-1}` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **≡** **{ }** **📄** Displaying documents 1 - 2 of 2 **<** **>** **REFRESH**

```

_id: ObjectId("62a17519f03ff532c125dec7")
title: " Star Trek Into Darkness "
year: 2013
> genre: Array
> actors: Array
> releases: Array

_id: ObjectId("62a17519f03ff532c125dec8")
title: " Iron Man 3 "
year: 2013
> genre: Array
> actors: Array

```

iv. Write a query that return the movie genre "Action" and "Adventure"

Hint use `{genre:{ $in:[" Action ", " Adventure "]}}`

