

- Importer le fichier json ci-joint dans votre base de données.
- Afficher les films de l'année 2010

```
> db.films.find({year:2010}).pretty()
[
  {
    _id: ObjectId("6009bbbf1833f4fecad0138"),
    title: '127 Hours',
    year: 2010,
    cast: [
      'James Franco',
      'Amber Tamblyn',
      'Clémence Poésy',
      'Lizzy Caplan'
    ],
    genres: [ 'Biography', 'Drama' ]
  },
  {
    _id: ObjectId("6009bbbf1833f4fecad0139"),
    title: '8: The Mormon Proposition',
    year: 2010,
    cast: [
      'The Church of Jesus Christ of Latter-day Saints',
      "' involvement in the 2008",
      'California Proposition 8',
      '. Narrated by',
      'Dustin Lance Black',
      '.'
    ],
    genres: [ 'Documentary' ]
  }
]
```

- Combien de film a été réalisé en 2009

```
> db.films.count({year:2009})
229
>
```

- Faites une requête pour récupérer des documents des films réalisés en 2008 en excluant les champs : `_id`, `cast`, `genres`

```
> db.films.find({year:2008},{_id:0, cast:0, genres:0}).pretty();
[
  { title: '10,000 BC', year: 2008 },
  { title: '21', year: 2008 },
  { title: '27 Dresses', year: 2008 },
  { title: '88 Minutes', year: 2008 },
  { title: 'The Accidental Husband', year: 2008 },
  { title: 'An American Carol', year: 2008 },
  { title: 'American Teen', year: 2008 },
  { title: 'Appaloosa', year: 2008 },
  { title: 'August', year: 2008 },
  { title: 'Baby Mama', year: 2008 },
  { title: 'Babylon A.D.', year: 2008 },
  { title: 'Ballast', year: 2008 },
  { title: 'Bangkok Dangerous', year: 2008 },
  { title: 'Be Kind Rewind', year: 2008 },
  { title: 'Bedtime Stories', year: 2008 },
  { title: 'Beer for My Horses', year: 2008 },
  { title: 'The Betrayal - Nerakhoun', year: 2008 },
  { title: 'Beverly Hills Chihuahua', year: 2008 },
  { title: 'Body of Lies', year: 2008 },
  { title: 'Bolt', year: 2008 }
]
```

- Afficher tous les films commençant par la lettre A

```
> db.films.find({title:{$regex:'^A'}}).pretty();
[
  {
    'Brian Bloom'
    _id: ObjectId("6009bbb8f1833f4fecac98b8"),
    title: 'After Dark in Central Park',
    year: 1900,
    cast: [],
    genres: [ ]tId("6009bbb8f1833f4fecad013b"),
  },title: 'A Little Help',
  { year: 2010,
    _id: ObjectId("6009bbb8f1833f4fecac98c9"),
    title: 'Acrobats in Cairo',
    year: 1901,Donnell",
    cast: [],nedict',
    genres: [ ]yrin',
  }, 'Daniel Yelsky'
  { ],
    _id: ObjectId("6009bbb8f1833f4fecac98ca"),
    title: 'An Affair of Honor',
    year: 1901,
    cast: [ ],ctId("6009bbb8f1833f4fecad013c"),
    genres: [ ]ventures of Power',
  },year: 2010,
  { cast: [
    _id: ObjectId("6009bbb8f1833f4fecac98cb"),
    title: 'Another Job for the Undertaker',
    year: 1901,ch',
```

- Afficher les films où 'Timothy Gibbs' a joué en 2011

```
db.films.find({$and:[{year:2011,cast:'Timothy Gibbs'}]}).pretty();
(
{
  'Violent J',
  _id: ObjectId("6009bbb8f1833f4fecad020a"),
  title: '11-11-11',
  year: 2011,,
  cast: [
    'Timothy Gibbs',
    'Michael Landes',
    'Wendy Glenn',
    'Benjamin Cook',
    'Lolo Herrero',
    'Salome Jimenez',
    'Brendan Price',, 'Western' ]
    'Denis Rafter',
    'Angela Rosal',
    'Lluís Soler'6009bbb8f1833f4fecad0146"),
  ],tle: 'Black Swan',
  genres: [ 'Horror', 'Thriller' ]
```

- Afficher les films de type 'Thriller' en 2011

```
> db.films.find({$and:[{year:2011,genres:'Thriller'}]}).pretty();
[
  {
    'Mila Kunis',
    {
      'Barbara Hershey',
      _id: ObjectId("6009bbbf1833f4fecad020a"),
      title: '11-11-11',
      year: 2011,Thriller' ]
      cast: [
        'Timothy Gibbs',
        'Michael Landes',bbbf1833f4fecad0147"),
        'Wendy Glenn',entine',
        'Benjamin Cook',
        'Lolo Herrero',ling', 'Michelle Williams' ],
        'Salome Jimenez',,, 'Drama' ]
        'Brendan Price',
        'Denis Rafter',
        'Angela Rosal',09bbbf1833f4fecad0148"),
        'Lluís Soler'k of Eli',
      ],ar: 2010,
      genres: [ 'Horror', 'Thriller' ]
    }, 'Denzel Washington',
```

- Afficher les films de type 'Thriller' réalisés en 2016 par ordre alphabétique inverse de leurs 'title'

```
> db.films.find({$and:[{year:2016,genres:'Thriller'}]}).sort({title:-1}).pretty();
[
  {
    {
      _id: ObjectId("6009bbbcf1833f4fecad06e4"),
      title: 'Triple 9',
      year: 2016,
      cast: [ 'Casey Affleck', 'Aaron Paul' ],
      genres: [ 'Crime', 'Thriller' ]
    },
    {
      _id: ObjectId("6009bbbcf1833f4fecad0719"),
      title: 'The Shallows',
      year: 2016,
      cast: [ 'Blake Lively' ],
      genres: [ 'Horror', 'Thriller' ]
    },
    {
```

- Insérer deux film de votre choix dans la base en utilisant un BulkWrite ( pour le champs 'year', il doit être 2020)

```
> db.films.bulkWrite(
...   [
...     { insertOne :
...       {
...         "document" :
...         {
...           "title" : "Anti-Life", "year" : 2020,
...         }
...       },
...     { insertOne :
...       {
...         "document" :
...         {
...           "title" : "Invisible Man", "year" : 2020,
...         }
...       }
...     }
...   ]
... );
{
  acknowledged: true,
  insertedCount: 2,
  insertedIds: {
    '0': ObjectId("6009c00ddf4c5f22ecddae1c"),
    '1': ObjectId("6009c00ddf4c5f22ecddae1d")
  },
}
```

- Supprimer tous les films réalisés avant 2000

```
> db.films.deleteMany( { year: { $lt: 2000 } } )
{ acknowledged: true, deletedCount: 24229 }
>
```

- Ajouter un champs rating qui sera de type array d'objets pour tous les documents

```
> db.films.update({},{$set: {"rating":[]}},false,true);
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- Faites vous et un collaborateur, les ratings (le rating est sur 5) de deux films de votre choix, comme suit :  
ratings: [ { by: "moi", rating: 4 }, {by:"collaborateur", rating: 5} ]

```

upsertedCount: 0
> db.films.update({ title: "Anti-Life" },{$push: {rating:{ $each: [ { by: "Mina", rating: 4 }, { by: "collaborateur", rating: 5 } ]}}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.films.update({ title: "Invisible Man" },{$push: {rating:{ $each: [ { by: "Mina", rating: 4 }, { by: "collaborateur", rating: 5 } ]}}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
>

```

- Créer un champ qui sera la moyenne de tous les ratings est appelé le : ar

```

db.films.aggregate([{$group:{ _id: "$title",ar: { $avg: "$rating" }}}]);
{ _id: 'Blow', ar: null },
{ _id: 'Deal', ar: null },
{ _id: 'I Saw the Light', ar: null },
{ _id: 'Ted 2', ar: null },
{ _id: 'Ant-Man and the Wasp', ar: null },
{ _id: 'The Texas Chainsaw Massacre', ar: null },
{ _id: 'Double Take', ar: null },
{ _id: 'Moving McAllister', ar: null },
{ _id: 'The Stepfather', ar: null },
{ _id: 'The Chaperone', ar: null },
{ _id: 'Saving Shiloh', ar: null },
{ _id: 'Miami Vice', ar: null },
{ _id: 'Pathology', ar: null },
{ _id: 'The Queen of Versailles', ar: null },
{ _id: 'Le Divorce', ar: null },
{ _id: 'Sinner', ar: null },
{ _id: 'High Fidelity', ar: null },
{ _id: 'Terminator Salvation', ar: null },
{ _id: 'The Ring', ar: null },
{ _id: 'Fade to Black', ar: null }

```

- Renommer le champ créer précédemment à savoir : ar, pour devenir averageRating

```
db.films.updateMany( {}, { $rename: { "ar": "averageRating" } } );

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4568,
  modifiedCount: 0,
  upsertedCount: 0
}
```

- Créer un champs views qui sera un array qui contiendra des valeurs comme ci-après : 'views':[123444, 66855,78966]

```
> db.films.update({},{$set: {"views":[123444, 66855,78966]}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
```

- Faites une mise à jour des films que vous avez insérer en renseignant les valeurs pour le tableau views.

```
db.films.update({}, {$set: {totalViews:""}}, false, true);

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- Créer un champs totalViews qui sera la somme du tableau view

```
> db.films.aggregate( [{
...     $group : {
...         _id : '$title',
...         totalViews : {$sum : '$views'}
...     }
... }
... ]
... );
[
  { _id: 'Apollo 18', totalViews: 0 },
  { _id: 'Iraq in Fragments', totalViews: 0 },
  { _id: 'Baby Boy', totalViews: 0 },
  { _id: 'Hatchet III', totalViews: 0 },
  { _id: 'Shallow Hal', totalViews: 0 },
  { _id: 'Code Name: The Cleaner', totalViews: 0 },
  { _id: 'Cloudy with a Chance of Meatballs', totalViews: 0 },
  { _id: 'Out Cold', totalViews: 0 },
  { _id: 'Sound of My Voice', totalViews: 0 },
  { _id: 'Secret of the Cave', totalViews: 0 },
  { _id: 'From Hell', totalViews: 0 },
  { _id: 'Quantum of Solace', totalViews: 0 },
  { _id: 'Dysfunctional Friends', totalViews: 0 },
  { _id: 'Going in Style', totalViews: 0 },
  { _id: 'Bumblebee', totalViews: 0 },
  { _id: 'K-PAX', totalViews: 0 },
  { _id: 'Live Freaky! Die Freaky!', totalViews: 0 },
  { _id: 'The Skulls', totalViews: 0 },
  { _id: 'Million Dollar Arm', totalViews: 0 },
  { _id: 'The Red Robin', totalViews: 0 }
]
```

- Créer un champs budget pour tous les documents et initialiser sa valeur à -1



```

]
> db.films.update({},{$set:{"budget":-1}},false,true);
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
>

```

- Modifier les valeurs budget pour les films que vous avez ajouté

```

> db.films.update({title : "Anti-Life"}, {$set:{ budget: 10}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.films.update({title : "Invisible Man"}, {$set:{ budget: 10}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
>

```