TP n°1, Exemples de systèmes NoSQL : clé-valeur (REDIS) et documentaires (MongoDB)

Ce travail pratique a pour objectif de découvrir les bases de données NoSQL à travers deux systèmes différents : Redis et MongoDB. Redis est utilisé comme base de données clé-valeur principalement en mémoire, tandis que MongoDB est une base de données orientée documents permettant de stocker et de manipuler des données au format JSON.

Partie 1: Installation et utilisation de REDIS

Les quatre grandes familles de bases de données NoSQL sont :

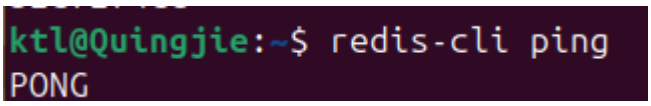
1. Clé–valeur → Redis, Riak
2. Colonnes → Cassandra, HBase
3. Documents → MongoDB, CouchDB
4. Graphes → Neo4j

Redis appartient à la famille clé–valeur : chaque donnée est identifiée par une clé unique associée à une valeur. Redis écoute par défaut l'adresse localhost (127.0.0.1) et le port 6379

Installation:

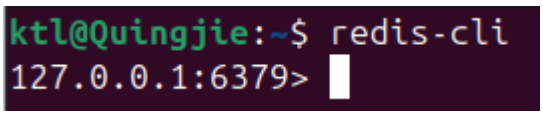
```
sudo apt update
sudo apt install redis-server

//Vérifier que Redis fonctionne
redis-server --version
redis-cli ping //si ça répond PONG, c'est bon
```



Ensuite, on démarre le serveur Redis avec: *redis-cli*. Si il y a une erreur lors du démarrage, cela dû au fait que le port 6379 (le port par défaut de Redis) est déjà occupé. Alors, il suffit de faire: *sudo systemctl stop redis-server* et de redémarrer le serveur Redis.

Une fois qu'on a démarré le serveur, on s'y connecte avec le client via un autre terminal grâce à la commande *redis-cli*.



1. Opération de Redis

- Create : SET
- Read : GET
- Update : SET
- Delete : DEL (si ça retourne 1, la clé est supprimée et si ça retourne 0 c'est que la clé est inexistante)

Commençons par créer une clé:

```
ktl@Quingjie:~$ redis-cli
127.0.0.1:6379> SET demo "Bonjour"
OK
127.0.0.1:6379>
```

Ici SET est la commande pour créer une clé, demo est son nom et Bonjour est la valeur qui lui est attribuée. La réponse OK signifie que la clé a bien été créée.

Pour lire une valeur, on fait:

```
127.0.0.1:6379> GET demo
"Bonjour"
```

Voici les autres opérations que nous pouvons faire:

```
127.0.0.1:6379> SET demo "salut"
OK
127.0.0.1:6379> DEL demo
(integer) 1
127.0.0.1:6379> GET demo
(nil)
```

La clé est stockée en mémoire RAM. Redis est souvent utilisé avec une base relationnelle pour assurer la persistance complète des données.

2. Compteurs

Créons un compteur:

```
127.0.0.1:6379> SET 15dec 0
OK
127.0.0.1:6379> INCR 15dec
(integer) 1
127.0.0.1:6379> GET 15dec
"1"
127.0.0.1:6379> INCR 15dec
(integer) 2
127.0.0.1:6379> GET 15dec
"2"
127.0.0.1:6379> DECR 15dec
(integer) 1
```

Chaque appel à INCR ajoute +1. Redis gère la concurrence : plusieurs utilisateurs peuvent incrémenter la même clé sans conflit.

3. Gestion de la durée de vie des clés

```
127.0.0.1:6379> SET macle valeur
OK
127.0.0.1:6379> TTL macle
(integer) -1
127.0.0.1:6379> EXPIRE macle 30
(integer) 1
127.0.0.1:6379> GET macle
"valeur"
127.0.0.1:6379> GET macle
(nil)
127.0.0.1:6379>
```

Ici on crée une clé mace. Avec la commande *TTL* on regarde son temps de vie (-1) durée infinie). Pour définir une expiration on utilise la commande *EXPIRE*, ici la commande indique que la clé sera supprimée après 30secs. Ici il y a eu moins de 30secs entre le *EXPIRE* et le 1er *GET*, d'où le fait qu'on a eu comme retour *"valeur"*, mais il y a eu plus de 30secs pour le 2eme, d'où le fait qu'on a eu comme retour *(nil)*.

4. Les listes

Pour insérer dans une liste des éléments on peut utiliser deux commandes: *LPUSH*, pour insérer à gauche, et *RPUSH*, pour insérer à droite.

```
127.0.0.1:6379> RPUSH mes_cours "BDA"
(integer) 1
127.0.0.1:6379> RPUSH mes_cours "Services Web"
(integer) 2
127.0.0.1:6379> LPUSH mes_cours "NoSQL"
(integer) 3
127.0.0.1:6379> LRANGE mes_cours 0 -1
1) "NoSQL"
2) "BDA"
3) "Services Web"
127.0.0.1:6379> LPOP mes_cours
"NoSQL"
127.0.0.1:6379> LRANGE mes_cours 0 -1
1) "BDA"
2) "Services Web"
```

La commande *LRANGE* permet d'afficher la liste avec 0 l'ordre du premier élément qu'on veut afficher et -1 l'ordre du dernier élément qu'on veut afficher (si on met 2 à la place on aura les 3 premiers éléments, -1 signifie le dernier élément de la liste)

```
127.0.0.1:6379> RPOP mes_cours
"Services Web"
```

La commande *RPOP* supprime l'élément à gauche.

5. Les ensembles

Un ensemble contient des éléments uniques, sans ordre.

La commande SADD permet d'ajouter des éléments. Lorsqu'on a un 1 cela signifie que l'élément est ajouté. Ici, on essaie d'ajouter deux fois l'utilisateur Samir, et lors de la requête on a 0 qui est retourné ce qui signifie que l'élément n'a pas été ajouté une deuxième fois: on ne peut pas avoir de doublons.

```
127.0.0.1:6379> SADD utilisateurs "Augustin"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Katell"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Samir"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Samir"
(integer) 0
127.0.0.1:6379> SMEMBERS utilisateurs
1) "Katell"
2) "Samir"
3) "Augustin"
127.0.0.1:6379> SREM utilisateurs "Samir"
(integer) 1
```

6. Union d’ensembles

```
127.0.0.1:6379> SADD autres_utilisateurs "Marc"
(integer) 1
127.0.0.1:6379> SADD autres_utilisateurs "Philippe"
(integer) 1
127.0.0.1:6379> SMEMBERS autres_utilisateurs
1) "Marc"
2) "Philippe"
127.0.0.1:6379> SUNION utilisateurs autres_utilisateurs
1) "Marc"
2) "Katell"
3) "Philippe"
4) "Augustin"
127.0.0.1:6379> SMEMBERS autres_utilisateurs
1) "Marc"
2) "Philippe"
127.0.0.1:6379> SMEMBERS utilisateurs
1) "Katell"
2) "Augustin"
```

La commande SUNION permet de faire l’union de deux ensembles sans avoir de doublons.

7. Ensembles ordonnés

Les ensembles ordonnés sont similaires aux ensembles (SET), mais chaque élément est associé à un score numérique. Les éléments sont uniques, mais triés automatiquement selon leur score.

Pour créer un ensemble de données, on utilise la commande *ZADD*.

```
127.0.0.1:6379> ZADD score 12 Samir
(integer) 1
127.0.0.1:6379> ZADD score 21 Katell
(integer) 1
127.0.0.1:6379> ZADD score 8 Marc
(integer) 1
127.0.0.1:6379> ZADD score 13 Augustin
(integer) 1
```

Pour trier les éléments dans l'ordre croissant, on utilise la commande *ZRANGE*, et pour les trier dans l'ordre décroissant on utilise la commande *ZREVRANGE*. Pour chacune on indique l'ordre du premier et du dernier élément qu'on veut avoir.

```
127.0.0.1:6379> ZRANGE score 0 -1
1) "Marc"
2) "Samir"
3) "Augustin"
4) "Katell"
127.0.0.1:6379> ZRANGE score 0 1
1) "Marc"
2) "Samir"
```

```
127.0.0.1:6379> ZREVRANGE score 0 -1
1) "Katell"
2) "Augustin"
3) "Samir"
4) "Marc"
```

On peut connaître l'ordre d'un élément avec la commande ZRANK:

```
127.0.0.1:6379> ZRANK score Augustin
(integer) 2
```

Dans la pratique :

- les scores sont stockés en base relationnelle ;
- Redis joue le rôle de cache en mémoire ;

- les classements sont calculés sans accès disque, ce qui améliore fortement les performances.

8. Mémoire et performances

Les bases relationnelles stockent les données sur disque. Chaque lecture implique un chargement en RAM. L'accès disque est beaucoup plus lent que l'accès mémoire.

Ordres de grandeur :

- RAM : ~10^{-8} secondes
- Disque : ~10^{-2} secondes

Redis stocke directement les données en RAM, ce qui explique ses performances élevées.

9. Les Hashes

Les hashes permettent de stocker une clé associée à plusieurs champs (clé → sous-clés → valeurs). C'est un peu comme un objet ou un dictionnaire.

Cas d'usage : représentation d'utilisateurs, données sans schéma fixe, distribution sur plusieurs nœuds.

Voici la création d'un hash avec la commande *HSET* champ par champ. La commande *HGETALL* permet de récupérer toutes les informations.

```
127.0.0.1:6379> HSET user:3 username Katell
(integer) 1
127.0.0.1:6379> HSET user:3 age 22
(integer) 1
```

```
127.0.0.1:6379> HSET user:3 email katell@edu.univ-paris13.fr
(integer) 1
127.0.0.1:6379> HGETALL user:3
1) "username"
2) "Katell"
3) "age"
4) "22"
5) "email"
6) "katell@edu.univ-paris13.fr"
```

On peut aussi créer un hash en une seule commande avec *HMSET*:

```
127.0.0.1:6379> HMSET user:4 username Augustin age 5 email augustin@gmail.fr
OK
127.0.0.1:6379> HGETALL user:4
1) "username"
2) "Augustin"
3) "age"
4) "5"
5) "email"
6) "augustin@gmail.fr"
127.0.0.1:6379> HINCRBY user:4 age 4
(integer) 9
127.0.0.1:6379> HGETALL user:4
1) "username"
2) "Augustin"
3) "age"
4) "9"
5) "email"
6) "augustin@gmail.fr"
```

La commande *HINCRBY* permet d'incrémenter une valeur.

```
127.0.0.1:6379> HVALS user:4
1) "Augustin"
2) "9"
3) "augustin@gmail.fr"
```

La commande *HVALS* permet d'avoir toutes les valeurs.

10. Publish/Subscribe

Le mécanisme Pub/Sub permet la communication temps réel entre plusieurs clients.

Cas d'usage : messagerie instantanée, notifications, diffusion d'événements.

Pour ce point, nous allons ouvrir un deuxième terminal dans lequel nous allons aussi démarrer le serveur Redis:

Le premier client souscrit à un canal avec la commande:

```
127.0.0.1:6379> SUBSCRIBE mes_cours
Reading messages... (press Ctrl-C to quit)
1) "subscribe"
2) "mes_cours"
3) (integer) 1
1) "message"
2) "mes_cours"
```

Il reste alors à l'écoute.

Lorsque le 2eme client publie quelque chose sur le canal:

```
ktl@Quingjie:~$ redis-cli
127.0.0.1:6379> PUBLISH mes_cours "Un nouveau cours sur MongoDB"
(integer) 1
127.0.0.1:6379>
```

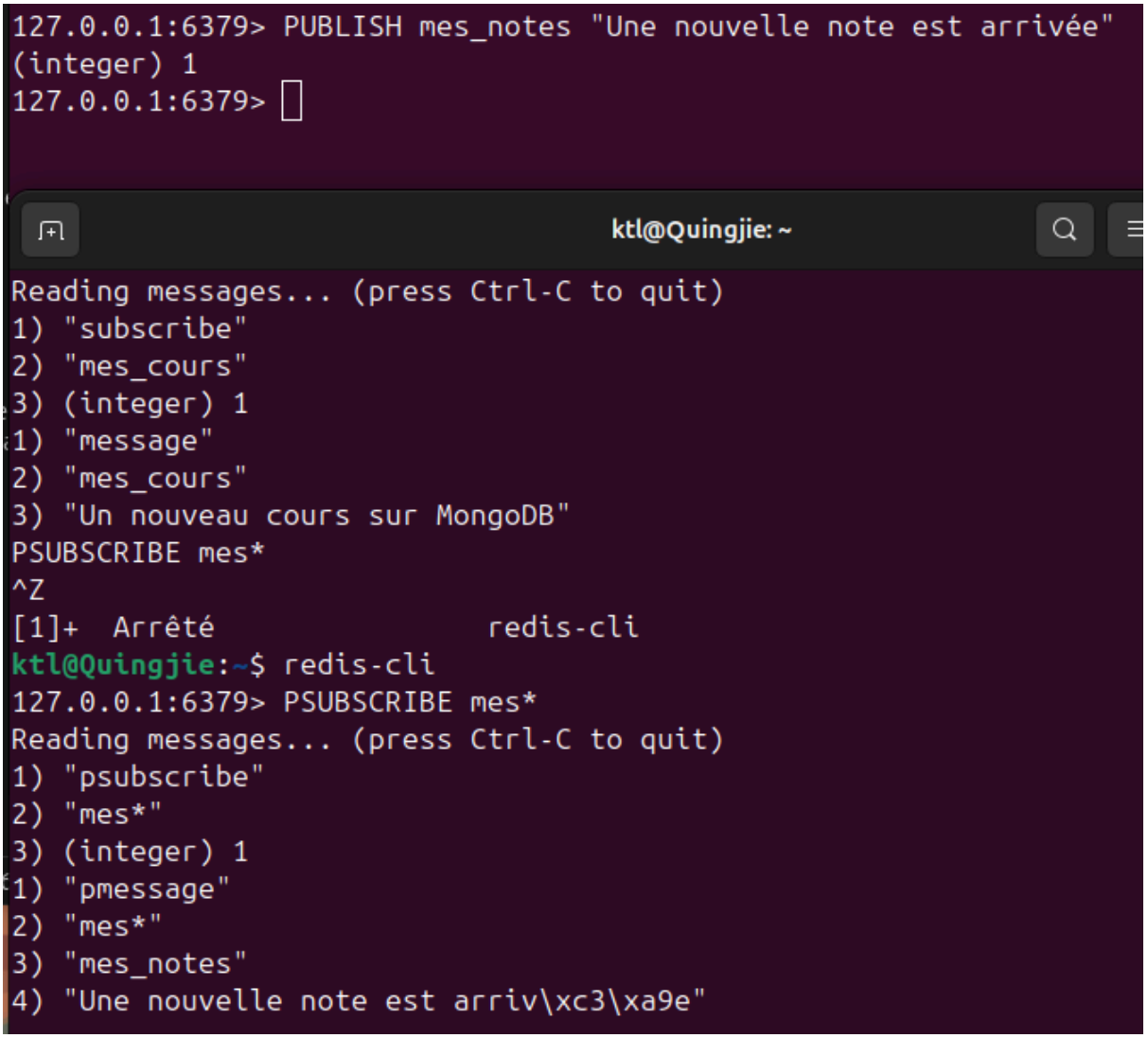
Tous les abonnés reçoivent le message en temps réel, dont le premier client:

```
127.0.0.1:6379> SUBSCRIBE mes_cours
Reading messages... (press Ctrl-C to quit)
1) "subscribe"
2) "mes_cours"
3) (integer) 1
1) "message"
2) "mes_cours"
3) "Un nouveau cours sur MongoDB"
```

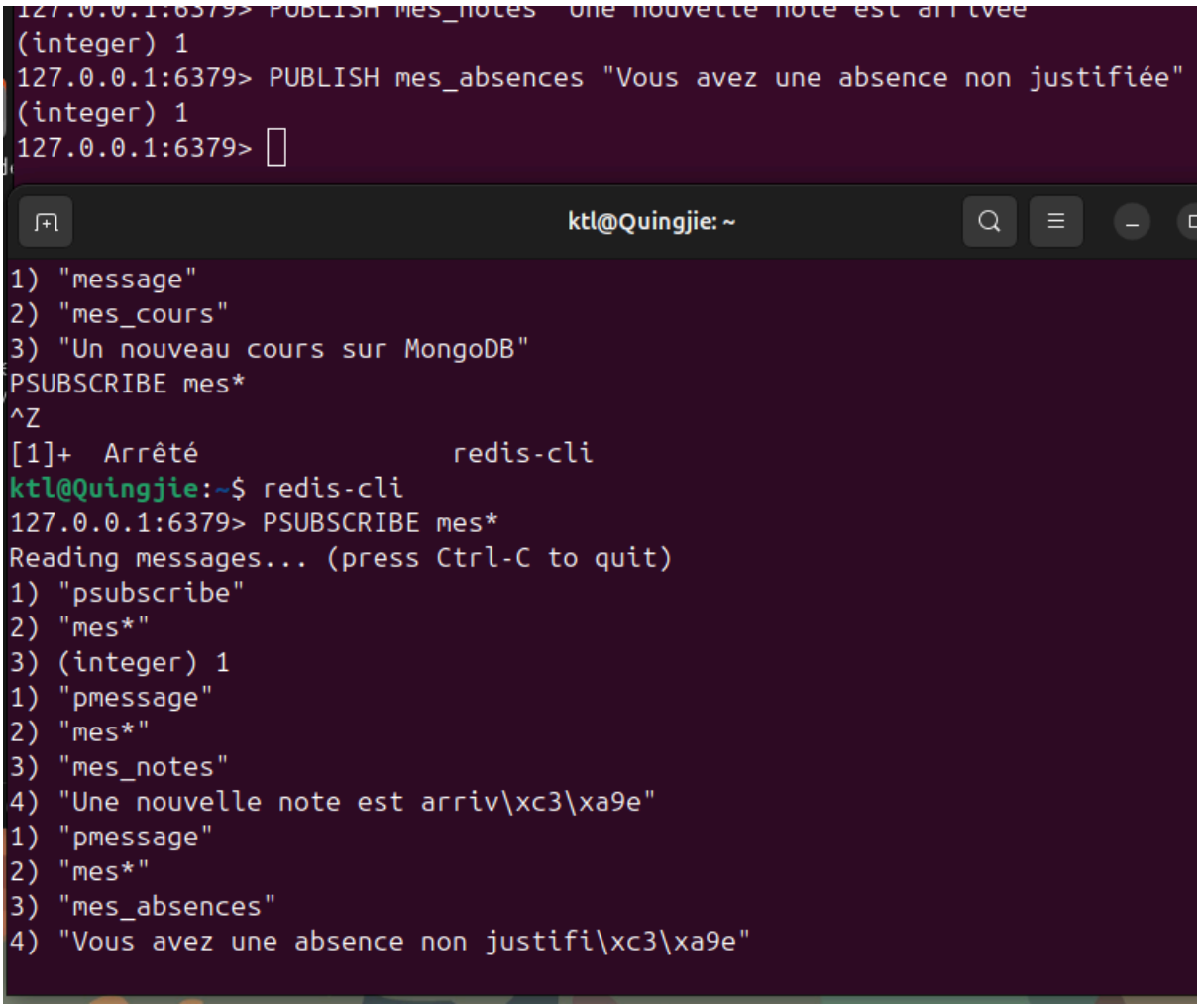
Le client peut aussi s'abonner à tous les canaux commençant par une chaines de caractères:

```
ktl@Quingjie:~$ redis-cli
127.0.0.1:6379> PSUBSCRIBE mes*
Reading messages... (press Ctrl-C to quit)
1) "psubscribe"
2) "mes*"
3) (integer) 1
```

Alors, lorsque dans le deuxième terminal on publie quelque chose, on a:



Et lorsqu’on publie un deuxième élément:



11. Gestion des bases de données Redis

Redis propose 16 bases de données par défaut (0 à 15).

On peut lister les clés avec la commande KEYS:

```
127.0.0.1:6379> KEYS *
 1) "user:4"
 2) "mes_cours"
 3) "score"
 4) "user:3"
 5) "mots"
 6) "score4"
 7) "15dec"
 8) "uer:3"
 9) "utilisateurs"
10) "autres_utilisateurs"
127.0.0.1:6379>
```

La commande *SELECT* permet de changer de base de données (ici la base de données 1 est vide. Pour revenir à la base de données par défaut, il suffit de faire *SELECT 0*.

```
127.0.0.1:6379> SELECT 1
OK
127.0.0.1:6379[1]> KEYS *
(empty array)
127.0.0.1:6379[1]> SELECT 0
OK
127.0.0.1:6379>
```

12. Persistance et précautions

Par défaut, Redis n'écrit pas immédiatement sur disque.

En cas de panne :

- certaines données peuvent être perdues ;
- la persistance doit être configurée explicitement.

Redis ne remplace pas totalement une base relationnelle, il la complète.

Partie 2: MongoDb

Installation de MongoDB via docker

```
docker pull mongo:7 //télécharger l'image officielle MongoDB
docker run -d --name mongodb -p 27017:27017 mongo:7 //lancer MongoDB dans un conteneur
docker ps //vérifier que MongoDB fonctionne
docker exec -it mongodb mongosh //se connecter à MongoDB
```

Ensuite, on sort de MongoDB et on télécharge les données avec la commande:

```
curl https://atlas-education.s3.amazonaws.com/sampledata.archive -o sampledata.archive
```

On copie le fichier dans le conteneur Docker:

```
docker cp sampledata.archive mongodb:/tmp/sampledata.archive
```

On a besoin de faire cela comme MongoDB tourne dans un conteneur Docker et que le fichier est lui sur l'ordinateur, mais pour l'utiliser avec mongodb, il faut qu'il soit dans le conteneur.

Ensuite il suffit de vérifier que le fichier est bien copié puis d'importer les données dans MongoDB, de se connecter à MongoDb et d'utiliser la base de données:

```
docker exec mongodb ls -lh /tmp/sampledata.archive
docker exec mongodb mongorestore --archive=/tmp/sampledata.archive
docker exec -it mongodb mongosh
use sample_mflix
```

```
test> use sample_mflix
switched to db sample_mflix
sample_mflix> show collections
comments
embedded_movies
movies
sessions
theaters
users
```

Maintenant il ne reste plus qu'à faire nos commandes:

1. Afficher les 5 films sortis depuis 2015.

```
db.movies.find({ year: { $gte: 2015 } }).limit(5)
```

```
sample_mflix> db.movies.find({ year: { $gte: 2015 } }).limit(5)
[
  {
    _id: ObjectId('573a13adf29313caabd2b765'),
    plot: "A new theme park is built on the original site of Jurassic Park. Everything is
going well until the park's newest attraction--a genetically modified giant stealth killin
g machine--escapes containment and goes on a killing spree.",
    genres: [ 'Action', 'Adventure', 'Sci-Fi' ],
    runtime: 124,
    metacritic: 59,
    rated: 'PG-13',
    cast: [
      'Chris Pratt',
      'Bryce Dallas Howard',
      'Irrfan Khan',
      "Vincent D'Onofrio"
    ],
    num_mflix_comments: 0,
    poster: 'https://m.media-amazon.com/images/M/MV5BNzQ3OTY4NjAtNzM5OS00N2ZhLWJlOWUtYzYwZ
jNmOWRiMzcyXkEyXkFqcGdeQXVyMTMxODk2OTU@._V1_SY1000_SX677_AL_.jpg',
    title: 'Jurassic World',
    fullplot: '22 years after the original Jurassic Park failed, the new park (also known
as Jurassic World) is open for business. After years of studying genetics the scientists o
n the park genetically engineer a new breed of dinosaur. When everything goes horribly wro
ng, will our heroes make it off the island?',
    languages: [ 'English' ],
    released: ISODate('2015-06-12T00:00:00.000Z'),
    directors: [ 'Colin Trevorrow' ],
    writers: [
      'Rick Jaffa (screenplay)',
      'Amanda Silver (screenplay)',
      'Colin Trevorrow (screenplay)',
      'Derek Connolly (screenplay)',
      'Rick Jaffa (story)',
      'Amanda Silver (story)',
      'Michael Crichton (characters)'
    ],
```

…

2. Trouver tous les films dont le genre est "Comedy".

```
db.movies.find({ genres: "Comedy" })
```

```
sample_mflix>  db.movies.find({ genres: "Comedy" })
[
  {
    _id: ObjectId('573a1390f29313caabcd4803'),
    plot: 'Cartoon figures announce, via comic strip balloons, that they will move - and r
ove they do, in a wildly exaggerated style.',
    genres: [ 'Animation', 'Short', 'Comedy' ],
    runtime: 7,
    cast: [ 'Winsor McCay' ],
    num_mflix_comments: 0,
    poster: 'https://m.media-amazon.com/images/M/MV5BYzg2NjNhNTctMjUxMi00ZWU4LWI3ZjYtNTI0M
TQxNThjZTk2XkEyXkFqcGdeQXVyNzg5OTk2OA@@._V1_SY1000_SX677_AL_.jpg',
    title: 'Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving Comics',
    fullplot: 'Cartoonist Winsor McCay agrees to create a large set of drawings that will
be photographed and made into a motion picture. The job requires plenty of drawing supplie
s, and the cartoonist must also overcome some mishaps caused by an assistant. Finally, the
 work is done, and everyone can see the resulting animated picture.',
    languages: [ 'English' ],
    released: ISODate('1911-04-08T00:00:00.000Z'),
    directors: [ 'Winsor McCay', 'J. Stuart Blackton' ],
    writers: [
      'Winsor McCay (comic strip "Little Nemo in Slumberland")',
      'Winsor McCay (screenplay)'
    ],
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-29 01:09:03.030000000',
    year: 1911,
    imdb: { rating: 7.3, votes: 1034, id: 1737 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.4, numReviews: 89, meter: 47 },
      lastUpdated: ISODate('2015-08-20T18:51:24.000Z')
    }
  },
  {
```

…

3. Afficher les sorties entre 2000 et 2005.

```
db.movies.find({year: {$gte: 2000, $lte: 2005}}, {title: 1, year: 1}).pretty()
```

```
sample_mflix> db.movies.find({year: {$gte: 2000, $lte: 2005}}, {title: 1, year: 1}).pretty
()
[
  {
    _id: ObjectId('573a1393f29313caabcdcb42'),
    title: 'Kate & Leopold',
    year: 2001
  },
  {
    _id: ObjectId('573a1398f29313caabceb1fe'),
    title: 'Crime and Punishment',
    year: 2002
  },
  {
    _id: ObjectId('573a139af29313caabcf0718'),
    year: 2001,
    title: 'Glitter'
  },
  {
    _id: ObjectId('573a139af29313caabcf0782'),
    year: 2000,
    title: 'In the Mood for Love'
  },
  {
    _id: ObjectId('573a139af29313caabcf0809'),
    year: 2003,
    title: 'The Manson Family'
  },
  {
    _id: ObjectId('573a139af29313caabcf0869'),
    title: 'The Dancer Upstairs',
    year: 2002
  },
  {
    _id: ObjectId('573a139af29313caabcf0d0b'),
    year: 2000,
    title: 'State and Main'
```

…

4. Afficher les films de genres “Drama” ET “Romance”.

```
db.movies.find({genres: {$all: ["Drama", "Romance"]}}, {title: 1, genres: 1})
```

```
sample_mflix> db.movies.find({genres: {$all: ["Drama", "Romance"]}}, {title: 1, genres:
... 1})
[
  {
    _id: ObjectId('573a1391f29313caabcd70b4'),
    genres: [ 'Drama', 'Romance', 'War' ],
    title: 'The Four Horsemen of the Apocalypse'
  },
  {
    _id: ObjectId('573a1391f29313caabcd7a34'),
    genres: [ 'Drama', 'Romance' ],
    title: 'A Woman of Paris: A Drama of Fate'
  },
  {
    _id: ObjectId('573a1391f29313caabcd7b98'),
    genres: [ 'Drama', 'Romance', 'Thriller' ],
    title: 'He Who Gets Slapped'
  },
  {
    _id: ObjectId('573a1391f29313caabcd7da6'),
    genres: [ 'Drama', 'Romance' ],
    title: 'Wild Oranges'
  },
  {
    _id: ObjectId('573a1391f29313caabcd89aa'),
    genres: [ 'Drama', 'Romance', 'War' ],
    title: 'Wings'
  },
  {
    _id: ObjectId('573a1391f29313caabcd91d7'),
    genres: [ 'Drama', 'Romance', 'Thriller' ],
    title: 'The Big House'
  },
  {
    _id: ObjectId('573a1391f29313caabcd9264'),
    genres: [ 'Romance', 'Drama' ],
    title: 'The Divorcee'
```

…

5. Afficher les films sans champ rated.

```
db.movies.find({rated: {$exists: false}}, {title: 1})
```

```
sample_mflix> db.movies.find({rated: {$exists: false}}, {title: 1})
[
  {
    _id: ObjectId('573a1390f29313caabcd4803'),
    title: 'Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving Comics'
  },
  {
    _id: ObjectId('573a1390f29313caabcd50e5'),
    title: 'Gertie the Dinosaur'
  },
  {
    _id: ObjectId('573a1390f29313caabcd516c'),
    title: 'In the Land of the Head Hunters'
  },
  {
    _id: ObjectId('573a1390f29313caabcd5293'),
    title: 'The Perils of Pauline'
  },
  { _id: ObjectId('573a1390f29313caabcd5a93'), title: 'Civilization' },
  {
    _id: ObjectId('573a1390f29313caabcd6223'),
    title: 'The Poor Little Rich Girl'
  },
  {
    _id: ObjectId('573a1390f29313caabcd6377'),
    title: 'Wild and Woolly'
  },
  { _id: ObjectId('573a1390f29313caabcd63d6'), title: 'The Blue Bird' },
  { _id: ObjectId('573a1391f29313caabcd6ea2'), title: 'The Saphead' },
  {
    _id: ObjectId('573a1391f29313caabcd71e3'),
    title: 'Miss Lulu Bett'
  },
  {
    _id: ObjectId('573a1391f29313caabcd72f0'),
    title: "Tol'able David"
  },
```

…

6. Afficher le nombre de films par année.

```
db.movies.aggregate([ {$group: {_id: "$year", total: {$sum: 1}}}, {$sort: {_id: 1}} ])
```

```
sample_mflix> db.movies.aggregate([
...   {$group: {_id: "$year", total: {$sum: 1}}},
...   {$sort: {_id: 1}}
... ])
[
  { _id: 1896, total: 2 }, { _id: 1903, total: 1 },
  { _id: 1909, total: 1 }, { _id: 1911, total: 2 },
  { _id: 1913, total: 1 }, { _id: 1914, total: 3 },
  { _id: 1915, total: 2 }, { _id: 1916, total: 2 },
  { _id: 1917, total: 2 }, { _id: 1918, total: 1 },
  { _id: 1919, total: 1 }, { _id: 1920, total: 4 },
  { _id: 1921, total: 5 }, { _id: 1922, total: 3 },
  { _id: 1923, total: 2 }, { _id: 1924, total: 6 },
  { _id: 1925, total: 3 }, { _id: 1926, total: 6 },
  { _id: 1927, total: 4 }, { _id: 1928, total: 8 }
]
Type "it" for more
sample_mflix>
```

7. Afficher la moyenne des notes IMDb par genre.

```
db.movies.aggregate([ {$unwind: "$genres"}, {$group: {_id: "$genres", moyenne: {$avg: "$imdb.rating"}}}, {$sort: {moyenne: -1}} ])
```

```
sample_mflix> db.movies.aggregate([
...   {$unwind: "$genres"},
...   {$group: {_id: "$genres", moyenne: {$avg: "$imdb.rating"}}},
...   {$sort: {moyenne: -1}}
... ])
[
  { _id: 'Film-Noir', moyenne: 7.397402597402598 },
  { _id: 'Short', moyenne: 7.377574370709382 },
  { _id: 'Documentary', moyenne: 7.365679824561403 },
  { _id: 'News', moyenne: 7.252272727272728 },
  { _id: 'History', moyenne: 7.1696100917431185 },
  { _id: 'War', moyenne: 7.128591954022989 },
  { _id: 'Biography', moyenne: 7.087984189723319 },
  { _id: 'Talk-Show', moyenne: 7 },
  { _id: 'Animation', moyenne: 6.89669603524229 },
  { _id: 'Music', moyenne: 6.883333333333334 },
  { _id: 'Western', moyenne: 6.823553719008264 },
  { _id: 'Drama', moyenne: 6.803377338624768 },
  { _id: 'Sport', moyenne: 6.749041095890411 },
  { _id: 'Crime', moyenne: 6.688585405625764 },
  { _id: 'Musical', moyenne: 6.665831435079727 },
  { _id: 'Romance', moyenne: 6.6564272782136396 },
  { _id: 'Mystery', moyenne: 6.527425044091711 },
  { _id: 'Adventure', moyenne: 6.493680884676145 },
  { _id: 'Comedy', moyenne: 6.450214658080344 },
  { _id: 'Fantasy', moyenne: 6.3829847908745245 }
]
```

8. Afficher le nombre de films par pays.

```
db.movies.aggregate([ {$unwind: "$countries"}, {$group: {_id: "$countries", total: {$sum: 1}}}, {$sort: {total: -1}} ])
```

```
sample_mflix> db.movies.aggregate([
...   {$unwind: "$countries"},
...   {$group: {_id: "$countries", total: {$sum: 1}}},
...   {$sort: {total: -1}}
... ])
[
  { _id: 'USA', total: 10921 },
  { _id: 'UK', total: 2652 },
  { _id: 'France', total: 2647 },
  { _id: 'Germany', total: 1494 },
  { _id: 'Canada', total: 1260 },
  { _id: 'Italy', total: 1217 },
  { _id: 'Japan', total: 786 },
  { _id: 'Spain', total: 675 },
  { _id: 'India', total: 564 },
  { _id: 'Australia', total: 470 },
  { _id: 'Sweden', total: 402 },
  { _id: 'Belgium', total: 364 },
  { _id: 'Hong Kong', total: 357 },
  { _id: 'Netherlands', total: 337 },
  { _id: 'Finland', total: 322 },
  { _id: 'Denmark', total: 308 },
  { _id: 'Russia', total: 290 },
  { _id: 'South Korea', total: 278 },
  { _id: 'China', total: 275 },
  { _id: 'West Germany', total: 246 }
]
```

9. Afficher les top 5 réalisateurs.

```
db.movies.aggregate([ {$unwind: "$directors"}, {$group: {_id: "$directors", total: {$sum: 1}}}, {$sort: {total: -1}}, {$limit: 5} ])
```

```
sample_mflix> db.movies.aggregate([
...   {$unwind: "$directors"},
...   {$group: {_id: "$directors", total: {$sum: 1}}},
...   {$sort: {total: -1}},
...   {$limit: 5}
... ])
[
  { _id: 'Woody Allen', total: 40 },
  { _id: 'Martin Scorsese', total: 32 },
  { _id: 'Takashi Miike', total: 31 },
  { _id: 'John Ford', total: 29 },
  { _id: 'Sidney Lumet', total: 29 }
]
```

10. Afficher les films triés par note IMDb.

```
db.movies.aggregate([ {$sort: {"imdb.rating": -1}}, {$project: {title: 1, "imdb.rating": 1}} ])
```

```
sample_mflix> db.movies.aggregate([
...   {$sort: {"imdb.rating": -1}},
...   {$project: {title: 1, "imdb.rating": 1}}
... ])
[
  {
    _id: ObjectId('573a1393f29313caabcddbed'),
    title: 'La nao capitana',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13b3f29313caabd3c7ac'),
    title: 'Landet som icke èr',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13b8f29313caabd4d540'),
    title: 'The Danish Girl',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13cef29313caabd86ddc'),
    title: 'Catching the Sun',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13cff29313caabd88f5b'),
    title: 'Scouts Guide to the Zombie Apocalypse',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13d3f29313caabd9473c'),
    title: 'No Tomorrow',
    imdb: { rating: '' }
  },
  {
    _id: ObjectId('573a13d3f29313caabd967ef')
```

…

11. Ajouter un champ état.

```
db.movies.updateOne({title: "Jaws"}, {$set: {etat: "culte"}})
```

```
Type "it" for more
sample_mflix> db.movies.updateOne({title: "Jaws"}, {$set: {etat: "culte"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

12. Incrémenter les votes IMDb de 100, par exemple.

```
db.movies.updateOne({title: "Inception"}, {$inc: {"imdb.votes": 100}})
```

```
sample_mflix> db.movies.updateOne({title: "Inception"}, {$inc: {"imdb.votes": 100}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

13. Supprimer le champ poster.

```
db.movies.updateMany({}, {$unset: {poster: ""}})
```

```
sample_mflix> db.movies.updateMany({}, {$unset: {poster: ""}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 21349,
  modifiedCount: 18044,
  upsertedCount: 0
}
```

14. Modifier le réalisateur

```
db.movies.updateOne({title: "Titanic"}, {$set: {directors: ["James Cameron"]}})
```

```
sample_mflix> db.movies.updateOne({title: "Titanic"}, {$set: {directors: ["James Cameron"]
}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

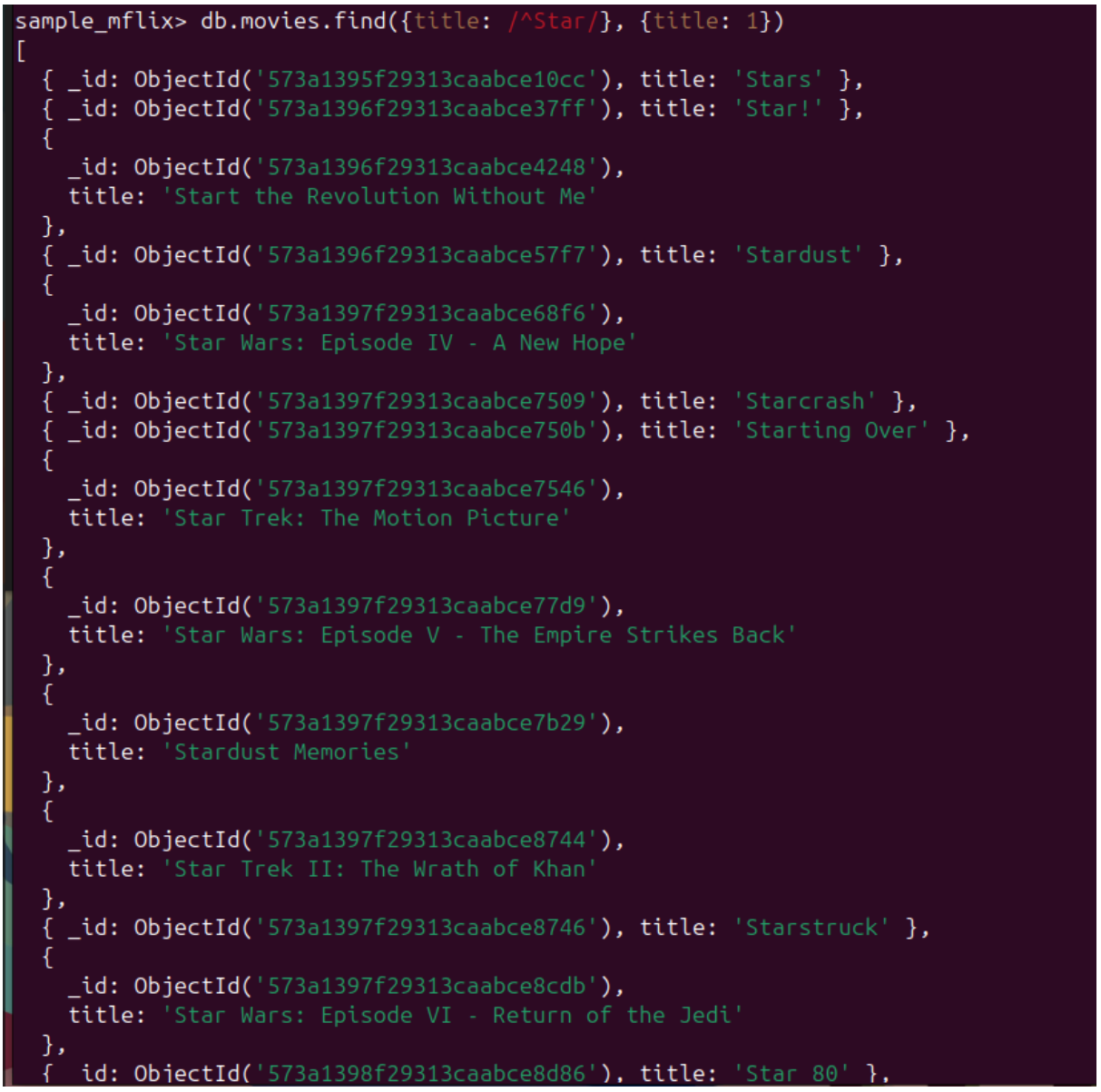
15. Afficher les films les mieux notés par décennie.

```
db.movies.aggregate([
  {$match: {"imdb.rating": {$exists: true}, year: {$type: "number"}}},
  {$project: {
    title: 1,
    decade: {$subtract: ["$year", {$mod: ["$year", 10]}]},
    "imdb.rating": 1
  }},
  {$group: {_id: "$decade", maxRating: {$max: "$imdb.rating"}}},
  {$sort: {_id: 1}}
])
```

```
sample_mflix> db.movies.aggregate([
...   {$match: {"imdb.rating": {$exists: true}, year: {$type: "number"}}},
...   {$project: {
...     title: 1,
...     decade: {$subtract: ["$year", {$mod: ["$year", 10]}]},
...     "imdb.rating": 1
...   }},
...   {$group: {_id: "$decade", maxRating: {$max: "$imdb.rating"}}},
...   {$sort: {_id: 1}}
... ])
[
  { _id: 1890, maxRating: 5.9 },
  { _id: 1900, maxRating: 7.4 },
  { _id: 1910, maxRating: 7.6 },
  { _id: 1920, maxRating: 8.3 },
  { _id: 1930, maxRating: 8.6 },
  { _id: 1940, maxRating: '' },
  { _id: 1950, maxRating: 8.6 },
  { _id: 1960, maxRating: 8.8 },
  { _id: 1970, maxRating: '' },
  { _id: 1980, maxRating: 9.3 },
  { _id: 1990, maxRating: 9.4 },
  { _id: 2000, maxRating: '' },
  { _id: 2010, maxRating: '' }
]
sample_mflix>
```

16. Afficher les films dont le titre commence par “Star”.

```
db.movies.find({title: /^Star/}, {title: 1})
```



```
sample_mflix> db.movies.find({title: /^Star/}, {title: 1})
[
  { _id: ObjectId('573a1395f29313caabce10cc'), title: 'Stars' },
  { _id: ObjectId('573a1396f29313caabce37ff'), title: 'Star!' },
  {
    _id: ObjectId('573a1396f29313caabce4248'),
    title: 'Start the Revolution Without Me'
  },
  { _id: ObjectId('573a1396f29313caabce57f7'), title: 'Stardust' },
  {
    _id: ObjectId('573a1397f29313caabce68f6'),
    title: 'Star Wars: Episode IV - A New Hope'
  },
  { _id: ObjectId('573a1397f29313caabce7509'), title: 'Starcrash' },
  { _id: ObjectId('573a1397f29313caabce750b'), title: 'Starting Over' },
  {
    _id: ObjectId('573a1397f29313caabce7546'),
    title: 'Star Trek: The Motion Picture'
  },
  {
    _id: ObjectId('573a1397f29313caabce77d9'),
    title: 'Star Wars: Episode V - The Empire Strikes Back'
  },
  {
    _id: ObjectId('573a1397f29313caabce7b29'),
    title: 'Stardust Memories'
  },
  {
    _id: ObjectId('573a1397f29313caabce8744'),
    title: 'Star Trek II: The Wrath of Khan'
  },
  { _id: ObjectId('573a1397f29313caabce8746'), title: 'Starstruck' },
  {
    _id: ObjectId('573a1397f29313caabce8cdb'),
    title: 'Star Wars: Episode VI - Return of the Jedi'
  },
  {  id: ObjectId('573a1398f29313caabce8d86'), title: 'Star 80' },
```

…

17. afficher les films avec plus de 2 genres.

```
db.movies.find({$where: "this.genres.length > 2"}, {title: 1, genres: 1})
```

```
sample_mflix> db.movies.find({$where: "this.genres.length > 2"}, {title: 1, genres: 1})
[
  {
    _id: ObjectId('573a1390f29313caabcd4803'),
    genres: [ 'Animation', 'Short', 'Comedy' ],
    title: 'Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving Comics'
  },
  {
    _id: ObjectId('573a1390f29313caabcd50e5'),
    genres: [ 'Animation', 'Short', 'Comedy' ],
    title: 'Gertie the Dinosaur'
  },
  {
    _id: ObjectId('573a1390f29313caabcd587d'),
    genres: [ 'Biography', 'Crime', 'Drama' ],
    title: 'Regeneration'
  },
  {
    _id: ObjectId('573a1390f29313caabcd6223'),
    genres: [ 'Comedy', 'Drama', 'Family' ],
    title: 'The Poor Little Rich Girl'
  },
  {
    _id: ObjectId('573a1390f29313caabcd6377'),
    genres: [ 'Comedy', 'Western', 'Romance' ],
    title: 'Wild and Woolly'
  },
  {
    _id: ObjectId('573a1391f29313caabcd68d0'),
    genres: [ 'Comedy', 'Short', 'Action' ],
    title: 'From Hand to Mouth'
  },
  {
    _id: ObjectId('573a1391f29313caabcd6f98'),
    genres: [ 'Crime', 'Drama', 'Mystery' ],
    title: 'The Ace of Hearts'
```

…

18. Afficher les films de Christopher Nolan.

```
db.movies.find({directors: "Christopher Nolan"}, {title: 1, year: 1, "imdb.rating": 1})
```

```
sample_mflix> db.movies.find({directors: "Christopher Nolan"}, {title: 1, year: 1,
... "imdb.rating": 1})
[
  {
    _id: ObjectId('573a139df29313caabcf8dd4'),
    imdb: { rating: 7.6 },
    year: 1998,
    title: 'Following'
  },
  {
    _id: ObjectId('573a13a0f29313caabd05acc'),
    imdb: { rating: 8.5 },
    year: 2000,
    title: 'Memento'
  },
  {
    _id: ObjectId('573a13a5f29313caabd1605f'),
    imdb: { rating: 7.2 },
    year: 2002,
    title: 'Insomnia'
  },
  {
    _id: ObjectId('573a13aef29313caabd2c349'),
    title: 'Batman Begins',
    year: 2005,
    imdb: { rating: 8.3 }
  },
  {
    _id: ObjectId('573a13b5f29313caabd42722'),
    imdb: { rating: 9 },
    year: 2008,
    title: 'The Dark Knight'
  },
  {
    _id: ObjectId('573a13b6f29313caabd45b78'),
    imdb: { rating: 8.5 },
    year: 2006,
```

…

19. Créer un index sur year

```
db.movies.createIndex({year: 1})
```

```
sample_mflix> db.movies.createIndex({year: 1})
year_1
```

20. Vérifier les index existants.

```
db.movies.getIndexes()
```

```
sample_mflix> db.movies.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  {
    v: 2,
    key: { _fts: 'text', _ftsx: 1 },
    name: 'cast_text_fullplot_text_genres_text_title_text',
    weights: { cast: 1, fullplot: 1, genres: 1, title: 1 },
    default_language: 'english',
    language_override: 'language',
    textIndexVersion: 3
  },
  { v: 2, key: { year: 1 }, name: 'year_1' }
]
```

21.. Comparer deux requêtes avec et sans index (utiliser explain()).

```
db.movies.find({year: 1995}).explain("executionStats")
```

```
sample_mflix> db.movies.find({year: 1995}).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'sample_mflix.movies',
    indexFilterSet: false,
    parsedQuery: { year: { '$eq': 1995 } },
    queryHash: '108EB0D0',
    planCacheKey: '5F26085F',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'FETCH',
      inputStage: {
        stage: 'IXSCAN',
        keyPattern: { year: 1 },
        indexName: 'year_1',
        isMultiKey: false,
        multiKeyPaths: { year: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { year: [ '[1995, 1995]' ] }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 372,
    executionTimeMillis: 1,
    totalKeysExamined: 372,
```

…

22. Supprimer l'index sur year.

```
db.movies.dropIndex({year: 1})
```

```
sample_mflix> db.movies.dropIndex({year: 1})
{ nIndexesWas: 3, ok: 1 }
sample_mflix>
```

23. créer un index composé sur year et imdb.rating .

```
db.movies.createIndex({year: 1, "imdb.rating": -1})
```

```
sample_mflix> db.movies.createIndex({year: 1, "imdb.rating": -1})
year_1_imdb.rating_-1
```

Partie 3:

On commence par importer les films avec la commande:

```
docker exec -i mongodb mongoimport \
  --db lesfilms \
  --collection films \
  --jsonArray \
  < films.json
```

```
ktl@Quingjie:~/Documents/Formation/Sup Galilée/Ing 3/NoSQL$ docker exec -i mongodb mongoim
port \
  --db lesfilms \
  --collection films \
  --jsonArray \
  < films.json
2025-12-16T16:21:46.134+0000    connected to: mongodb://localhost/
2025-12-16T16:21:46.190+0000    278 document(s) imported successfully. 0 document(s) faile
d to import.
```

Ensuite on rentre dans MongoDB avec la commande *docker exec -it mongodb mongosh*.

1. Vérifier que les données ont été importées, en les comptant par exemple.

```
test> use lesfilms
switched to db lesfilms
lesfilms> db.films.countDocuments()
278
```

2. Avant de commencer à interroger la base de données des films, il est essentiel de bien comprendre la structure d'un document dans notre collection de films. Pour ce faire, nous allons utiliser la fonction findOne(), qui permet de récupérer un seul document.

```
lesfilms> db.films.findOne()
...
{
  _id: 'movie:11',
  title: 'La Guerre des étoiles',
  year: 1977,
  genre: 'Aventure',
  summary: "Il y a bien longtemps, dans une galaxie très lointaine... La guerre civile fai
t rage entre l'Empire galactique et l'Alliance rebelle. Capturée par les troupes de choc d
e l'Empereur menées par le sombre et impitoyable Dark Vador, la princesse Leia Organa diss
imule les plans de l'Étoile Noire, une station spatiale invulnérable, à son droïde R2-D2 a
vec pour mission de les remettre au Jedi Obi-Wan Kenobi. Accompagné de son fidèle compagno
n, le droïde de protocole C-3PO, R2-D2 s'échoue sur la planète Tatooine et termine sa quêt
e chez le jeune Luke Skywalker. Rêvant de devenir pilote mais confiné aux travaux de la fe
rme, ce dernier se lance à la recherche de ce mystérieux Obi-Wan Kenobi, devenu ermite au
cœur des montagnes désertiques de Tatooine...",
  country: 'US',
  director: {
    _id: 'artist:1',
    last_name: 'Lucas',
    first_name: 'George',
    birth_date: 1944
  },
  actors: [
    { last_name: 'Hamill', first_name: 'Mark', birth_date: 1951 },
    { last_name: 'Ford', first_name: 'Harrison', birth_date: 1942 },
    { last_name: 'Fisher', first_name: 'Carrie', birth_date: 1956 },
    { last_name: 'Cushing', first_name: 'Peter', birth_date: 1913 },
    { last_name: 'Daniels', first_name: 'Anthony', birth_date: 1946 },
    { last_name: 'Earl Jones', first_name: 'James', birth_date: 1931 },
    { last_name: 'Prowse', first_name: 'David', birth_date: 1935 },
    { last_name: 'Mayhew', first_name: 'Peter', birth_date: 1944 }
  ],
  grades: [
    { note: 93, grade: 'C' },
    { note: 68, grade: 'C' },
    { note: 50, grade: 'E' },
    { note: 29, grade: 'D' }
  ]
}
```

3. Afficher la liste des films d’action.

```
lesfilms> db.films.find({ genre: "Action" })
[
  {
    _id: 'movie:24',
    title: 'Kill Bill : Volume 1',
    year: 2003,
    genre: 'Action',
    summary: "Au cours d'une cérémonie de mariage en plein désert, un commando fait irr
ion dans la chapelle et tire sur les convives. Laissée pour morte, la Mariée enceinte r
ouve ses esprits après un coma de quatre ans. Celle qui a auparavant exercé les fonctio
de tueuse à gages au sein du Détachement International des Vipères Assassines n'a alors
us qu'une seule idée en tête : venger la mort de ses proches en éliminant tous les memb
 de l'organisation criminelle, dont leur chef Bill qu'elle se réserve pour la fin.",
    country: 'US',
    director: {
      _id: 'artist:138',
      last_name: 'Tarantino',
      first_name: 'Quentin',
      birth_date: 1963
    },
    actors: [
      { last_name: 'Thurman', first_name: 'Uma', birth_date: 1970 },
      { last_name: 'Liu', first_name: 'Lucy', birth_date: 1968 },
      { last_name: 'Carradine', first_name: 'David', birth_date: 1936 },
      { last_name: 'Madsen', first_name: 'Michael', birth_date: 1957 },
      { last_name: 'Hannah', first_name: 'Daryl', birth_date: 1960 },
      { last_name: 'A. Fox', first_name: 'Vivica', birth_date: 1964 },
      { last_name: 'Parks', first_name: 'Michael', birth_date: 1940 },
      { last_name: 'Chiba', first_name: 'Sonny', birth_date: 1939 },
      { last_name: 'Kuriyama', first_name: 'Chiaki', birth_date: 1984 },
      { last_name: 'Dreyfus', first_name: 'Julie', birth_date: 1966 },
      {
        last_name: 'Liu Chia-Hui',
        first_name: 'Gordon',
        birth_date: 1955
      }
    ],
    grades: [
      { note: 83, grade: 'E' },
      { note: 65, grade: 'B' },
      { note: 48, grade: 'B' },
      { note: 84, grade: 'A' }
```

4. Afficher le nombre de films d'action.

```
lesfilms> db.films.countDocuments({ genre: "Action" })
36
lesfilms>
```

5. Afficher les films d'action produits en France

```
lesfilms> db.films.find({
...   genre: "Action",
...   country: "FR"
... })
...
[
  {
    _id: 'movie:4034',
    title: "L'Homme de Rio",
    year: 1964,
    genre: 'Action',
    summary: "Le deuxième classe Adrien Dufourquet est témoin de l'enlèvement de sa fiancé
e Agnès, fille d'un célèbre ethnologue. Il part à sa recherche, qui le mène au Brésil, et
met au jour un trafic de statuettes indiennes.",
    country: 'FR',
    director: {
      _id: 'artist:34613',
      last_name: 'de Broca',
      first_name: 'Philippe',
      birth_date: 1933
    },
    actors: [
      {
        last_name: 'Belmondo',
        first_name: 'Jean-Paul',
        birth_date: 1933
      },
      { last_name: 'Celi', first_name: 'Adolfo', birth_date: 1922 },
      { last_name: 'Servais', first_name: 'Jean', birth_date: 1910 },
      {
        last_name: 'Dorléac',
        first_name: 'Françoise',
        birth_date: 1942
      },
      { last_name: 'Renant', first_name: 'Simone', birth_date: 1911 }
    ],
    grades: [
      { note: 4, grade: 'D' },
      { note: 30, grade: 'E' },
      { note: 34, grade: 'E' },
      { note: 28, grade: 'F' }
```

…

6. Afficher les films d’action produits en France en 1963.

```
lesfilms> db.films.find({
...   genre: "Action",
...   country: "FR",
...   year: 1963
... })
...
[
  {
    _id: 'movie:25253',
    title: 'Les tontons flingueurs',
    year: 1963,
    genre: 'Action',
    summary: "Sur son lit de mort, le Mexicain fait promettre à son ami d'enfance, Fernand
 Naudin, de veiller sur ses intérêts et sa fille Patricia. Fernand découvre alors qu'il se
 trouve à la tête d'affaires louches dont les anciens dirigeants entendent bien s'emparer.
 Mais, flanqué d'un curieux notaire et d'un garde du corps, Fernand impose d'emblée sa loi
. Cependant, le belle Patricia lui réserve quelques surprises !",
    country: 'FR',
    director: {
      _id: 'artist:18563',
      last_name: 'Lautner',
      first_name: 'Georges',
      birth_date: 1926
    },
    actors: [
      { last_name: 'Ventura', first_name: 'Lino', birth_date: 1919 },
      { last_name: 'Frank', first_name: 'Horst', birth_date: 1929 },
      { last_name: 'Dalban', first_name: 'Robert', birth_date: 1903 },
      { last_name: 'Blier', first_name: 'Bernard', birth_date: 1916 },
      { last_name: 'Rich', first_name: 'Claude', birth_date: 1929 },
      { last_name: 'Sinjen', first_name: 'Sabine', birth_date: 1942 },
      { last_name: 'Lefebvre', first_name: 'Jean', birth_date: 1920 },
      { last_name: 'Bertin', first_name: 'Pierre', birth_date: 1891 },
      { last_name: 'Blanche', first_name: 'Francis', birth_date: 1919 }
    ],
    grades: [
      { note: 35, grade: 'C' },
      { note: 48, grade: 'F' },
      { note: 64, grade: 'C' },
      { note: 42, grade: 'F' }
    ]
  }
]
```

7. Jusqu'à maintenant, à chaque fois qu'un document est renvoyé, tous ses attributs sont affichés. Pour n'afficher que certains attributs, on utilise un filtre. Pour afficher les films d'action réalisés en France, sans les grades, le filtre est passé comme deuxième paramètre de la fonction find() comme ceci.

```
lesfilms> db.films.find(
...   {
...     genre: "Action",
...     country: "FR"
...   },
...   {
...     grades: 0
...   }
... )
...
[
  {
    _id: 'movie:4034',
    title: "L'Homme de Rio",
    year: 1964,
    genre: 'Action',
    summary: "Le deuxième classe Adrien Dufourquet est témoin de l'enlèvement de sa fiancé
e Agnès, fille d'un célèbre ethnologue. Il part à sa recherche, qui le mène au Brésil, et
met au jour un trafic de statuettes indiennes.",
    country: 'FR',
    director: {
      _id: 'artist:34613',
      last_name: 'de Broca',
      first_name: 'Philippe',
      birth_date: 1933
    },
    actors: [
      {
        last_name: 'Belmondo',
        first_name: 'Jean-Paul',
        birth_date: 1933
      },
      { last_name: 'Celi', first_name: 'Adolfo', birth_date: 1922 },
      { last_name: 'Servais', first_name: 'Jean', birth_date: 1910 },
      {
        last_name: 'Dorléac',
        first_name: 'Françoise',
        birth_date: 1942
      },
      { last_name: 'Renant', first_name: 'Simone', birth_date: 1911 }
    ]
  },
  {
```

8. Vous avez certainement remarqué que les identifiants des documents sont affichés même si un filtre est appliqué aux résultats. Pour ne pas les afficher, vous pouvez ajouter sa valeur est la mettre à zéro, comme ceci (cette requête e nous renvois tous les films d’action r´ealis´es en France sans les

identifiants)

```
lesfilms> db.films.find(
...   {
...     genre: "Action",
...     country: "FR"
...   },
...   {
...     grades: 0,
...     _id: 0
...   }
... )
...
[
  {
    title: "L'Homme de Rio",
    year: 1964,
    genre: 'Action',
    summary: "Le deuxième classe Adrien Dufourquet est témoin de l'enlèvement de sa fiancé
e Agnès, fille d'un célèbre ethnologue. Il part à sa recherche, qui le mène au Brésil, et
met au jour un trafic de statuettes indiennes.",
    country: 'FR',
    director: {
      _id: 'artist:34613',
      last_name: 'de Broca',
      first_name: 'Philippe',
      birth_date: 1933
    },
    actors: [
      {
        last_name: 'Belmondo',
        first_name: 'Jean-Paul',
        birth_date: 1933
      },
      { last_name: 'Celi', first_name: 'Adolfo', birth_date: 1922 },
      { last_name: 'Servais', first_name: 'Jean', birth_date: 1910 },
      {
        last_name: 'Dorléac',
        first_name: 'Françoise',
        birth_date: 1942
      },
      { last_name: 'Renant', first_name: 'Simone', birth_date: 1911 }
    ]
  },
  {
```

…

9. Afficher les titres et les grades des films d’action réalisés en France sans leurs identifiants.

```
lesfilms> db.films.find(
...   {
...     genre: "Action",
...     country: "FR"
...   },
...   {
...     title: 1,
...     grades: 1,
...     _id: 0
...   }
... )
...
[
  {
    title: "L'Homme de Rio",
    grades: [
      { note: 4, grade: 'D' },
      { note: 30, grade: 'E' },
      { note: 34, grade: 'E' },
      { note: 28, grade: 'F' }
    ]
  },
  {
    title: 'Nikita',
    grades: [
      { note: 88, grade: 'F' },
      { note: 36, grade: 'C' },
      { note: 74, grade: 'D' },
      { note: 62, grade: 'D' }
    ]
  },
  {
    title: 'Les tontons flingueurs',
    grades: [
      { note: 35, grade: 'C' },
      { note: 48, grade: 'F' },
      { note: 64, grade: 'C' },
      { note: 42, grade: 'F' }
    ]
  }
]
lesfilms>
```

10. Les filtres affichés jusqu'à présent sont par valeur exacte. Afficher les titres et les notes des films d'action réalisés en France et ayant obtenu une note supérieure à 10. Remarquez que même si des films ont obtenu certaines notes inférieures à 10 sont affichés.

```
lesfilms> db.films.find(
...   {
...     genre: "Action",
...     country: "FR",
...     "grades.grade": { $gt: 10 }
...   },
...   {
...     title: 1,
...     grades: 1,
...     _id: 0
...   }
... )
...

lesfilms>
```

11. Si l'on souhaite afficher que des films ayant des notes supérieures à 10, on doit préciser que les notes obtenues sont toutes supérieures à 40. La requête suivante permet d'afficher les films d'action réalisés en France ayant obtenus que des notes supérieures à 10.

```
lesfilms> db.films.find(
...   {
...     genre: "Action",
...     country: "FR",
...     grades: { $not: { $elemMatch: { grade: { $lte: 10 } } } }
...   }
... )
...
[
  {
    _id: 'movie:4034',
    title: "L'Homme de Rio",
    year: 1964,
    genre: 'Action',
    summary: "Le deuxième classe Adrien Dufourquet est témoin de l'enlèvement de sa fiancé
e Agnès, fille d'un célèbre ethnologue. Il part à sa recherche, qui le mène au Brésil, et
met au jour un trafic de statuettes indiennes.",
    country: 'FR',
    director: {
      _id: 'artist:34613',
      last_name: 'de Broca',
      first_name: 'Philippe',
      birth_date: 1933
    },
    actors: [
      {
        last_name: 'Belmondo',
        first_name: 'Jean-Paul',
        birth_date: 1933
      },
      { last_name: 'Celi', first_name: 'Adolfo', birth_date: 1922 },
      { last_name: 'Servais', first_name: 'Jean', birth_date: 1910 },
      {
        last_name: 'Dorléac',
        first_name: 'Françoise',
        birth_date: 1942
      },
      { last_name: 'Renant', first_name: 'Simone', birth_date: 1911 }
    ],
    grades: [
      { note: 4, grade: 'D' },
      { note: 30, grade: 'E' },
```

12. Afficher les différents genres de base lesfilms.

```
lesfilms> db.films.distinct("genre")
[
  'Action',          'Adventure',
  'Aventure',        'Comedy',
  'Comédie',         'Crime',
  'Drama',           'Drame',
  'Fantastique',     'Fantasy',
  'Guerre',          'Histoire',
  'Horreur',         'Musique',
  'Mystery',         'Mystère',
  'Romance',         'Science Fiction',
  'Science-Fiction', 'Thriller',
  'War',             'Western'
]
```

13. Afficher les différents grades attribués

```
lesfilms> db.films.distinct("grades.grade")
...
[ 'A', 'B', 'C', 'D', 'E', 'F' ]
```

14. Afficher tous les films dans lesquels joue au moins un des artistes suivants [”artist:4”,”artist:18”,”artist:11”].

```
lesfilms> db.films.aggregate([
...   { $unwind: "$actors" },
...   { $project: {
...       fields: { $objectToArray: "$actors" }
...   }},
...   { $unwind: "$fields" },
...   { $group: {
...       _id: "$fields.k"
...   }},
...   { $sort: { _id: 1 } }
... ])
[ { _id: 'birth_date' }, { _id: 'first_name' }, { _id: 'last_name' } ]
lesfilms>
```

Ce n’est pas possible, car il n’y a pas d’identifiant pour les acteurs, et il n’y a pas de champs artist. Par contre c’est possible pour les producteurs:

```
lesfilms> db.films.find({
...   "director._id": { $in: ["artist:4", "artist:18", "artist:11"] }
... }, {
...   title: 1,
...   year: 1,
...   director: 1
... })

lesfilms>
```

15. Afficher tous les films qui n'ont pas de résumés.

```
lesfilms> db.films.find({
...   summary: { $exists: false }
... })
...

lesfilms>
```

Tous les films ont un résumé.

16. Afficher tous les films tournés avec Leonardo DiCaprio en 1997.

```
lesfilms> db.films.find({
...   year: 1997,
...   "actors.last_name": "DiCaprio",
...   "actors.first_name": "Leonardo"
... })
...
[
  {
    _id: 'movie:597',
    title: 'Titanic',
    year: 1997,
    genre: 'Drame',
    summary: 'Southampton, 10 avril 1912. Le paquebot le plus grand et le plus moderne du
monde, réputé pour son insubmersibilité, le « Titanic », appareille pour son premier voyag
e. 4 jours plus tard, il heurte un iceberg. À son bord, un artiste pauvre et une grande bo
urgeoise tombent amoureux.',
    country: 'US',
    director: {
      _id: 'artist:2710',
      last_name: 'Cameron',
      first_name: 'James',
      birth_date: 1954
    },
    actors: [
      { last_name: 'Winslet', first_name: 'Kate', birth_date: 1975 },
      { last_name: 'Zane', first_name: 'Billy', birth_date: 1966 },
      { last_name: 'Fisher', first_name: 'Frances', birth_date: 1952 },
      {
        last_name: 'DiCaprio',
        first_name: 'Leonardo',
        birth_date: 1974
      },
      { last_name: 'Bates', first_name: 'Kathy', birth_date: 1948 },
      { last_name: 'Stuart', first_name: 'Gloria', birth_date: 1910 }
    ],
    grades: [
      { note: 40, grade: 'C' },
      { note: 91, grade: 'B' },
      { note: 45, grade: 'D' },
      { note: 7, grade: 'A' }
    ]
  }
]
```

17. Afficher les films tournés avec Leonardo DiCaprio ou 1997.

```
lesfilms> db.films.find({
...   $or: [
...     { "actors.last_name": "DiCaprio", "actors.first_name": "Leonardo" },
...     { year: 1964 }
...   ]
... })
...
[
  {
    _id: 'movie:597',
    title: 'Titanic',
    year: 1997,
    genre: 'Drame',
    summary: 'Southampton, 10 avril 1912. Le paquebot le plus grand et le plus moderne du
monde, réputé pour son insubmersibilité, le « Titanic », appareille pour son premier voyag
e. 4 jours plus tard, il heurte un iceberg. À son bord, un artiste pauvre et une grande bo
urgeoise tombent amoureux.',
    country: 'US',
    director: {
      _id: 'artist:2710',
      last_name: 'Cameron',
      first_name: 'James',
      birth_date: 1954
    },
    actors: [
      { last_name: 'Winslet', first_name: 'Kate', birth_date: 1975 },
      { last_name: 'Zane', first_name: 'Billy', birth_date: 1966 },
      { last_name: 'Fisher', first_name: 'Frances', birth_date: 1952 },
      {
        last_name: 'DiCaprio',
        first_name: 'Leonardo',
        birth_date: 1974
      },
      { last_name: 'Bates', first_name: 'Kathy', birth_date: 1948 },
      { last_name: 'Stuart', first_name: 'Gloria', birth_date: 1910 }
    ],
    grades: [
      { note: 40, grade: 'C' },
      { note: 91, grade: 'B' },
      { note: 45, grade: 'D' },
      { note: 7, grade: 'A' }
    ]
```

…