Rapport : No SQL - TP1

Introduction a Redis et MongoDB

redis

mongoDB®

ERRACHDI Ilias

3 ème année informatique AP

Année scolaire
2025-2026

Introduction aux bases de données NoSQL et à Redis

Comparaison entre bases de données relationnelles et NoSQL

Pendant longtemps, les bases de données relationnelles (SGBDR) ont été le standard pour le stockage et la gestion des données. Elles reposent sur une structure stricte composée de tables et de relations, et utilisent le langage SQL pour les requêtes. Les SGBDR garantissent les propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité), assurant ainsi la fiabilité et la cohérence des transactions, même en cas de défaillance système.

Les bases de données NoSQL se répartissent en plusieurs types selon leur modèle de stockage et leur méthode d'interrogation :

_ **Bases Clé-Valeu**r : Stockent les données sous forme de paires clé-valeur, permettant un accès très rapide.

Exemples : Redis, DynamoDB.

_ **Bases Orientées Colonnes** : Organisent les données par colonnes plutôt que par lignes, ce qui optimise les requêtes analytiques sur de grands volumes de données.

Exemples : Apache Cassandra, HBase.

_ **Bases Orientées Documents** : Stockent les informations sous forme de documents JSON ou BSON, offrant une grande flexibilité dans la structuration des données.

Exemples : MongoDB, CouchDB.

_ **Bases Orientées Graphes** : Conçues pour représenter des relations complexes entre les données via des graphes de nœuds et d'arêtes.

Exemples : Neo4j, FlockDB.

_ **Bases Série Temporelle** : Optimisées pour la gestion de données chronologiques et les requêtes basées sur le temps.

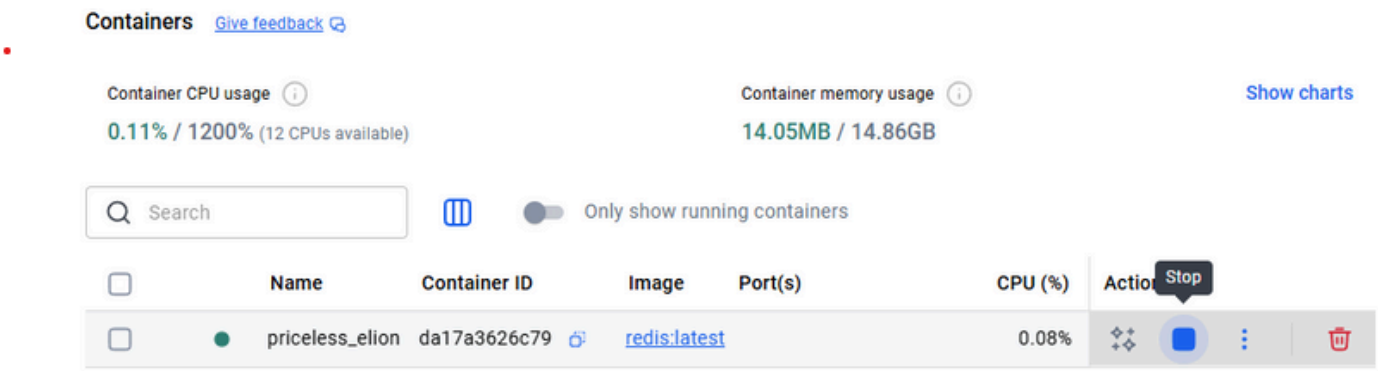
Exemples : InfluxDB, OpenTSDB.

Tutoriel sur Redis : Manipulation des Clés et Structures de Données

Telechargement de l'image redis

```
PS C:\Users\ilias> docker pull redis
Using default tag: latest
latest: Pulling from library/redis
b4c7acd54b97: Pull complete
5325bfd41068: Pull complete
8e44f01296e3: Pull complete
4f4fb700ef54: Pull complete
98ac9c138461: Pull complete
5d05fbd0b692: Pull complete
8116b2f58ddb: Pull complete
Digest: sha256:43355efd22490e31ca14b9d569367d05121e2be61fd8e47937563ae2a80952ae
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
```

Demarer le contoneur docker depuis Docker desktop




```
PS C:\Users\ing> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                    NAMES
303fffaf78a3   redis:latest   "docker-entrypoint.s…"   5 minutes ago   Up 5 minutes   6379/tcp                 busy_hopper
5dcb22b8b7ae   redis          "docker-entrypoint.s…"   7 minutes ago   Up 7 minutes   0.0.0.0:6379->6379/tcp   redis
```

Demarer le client

```
PS C:\Users\ing> docker exec -it redis redis-cli
```

Execution de certain commande REDIS

```
127.0.0.1:6379> SET demo "Bonjour"
OK
127.0.0.1:6379> set user:1234 "Ilias"
OK
127.0.0.1:6379> get user:1234
"Ilias"
127.0.0.1:6379> del user:12346
(integer) 0
127.0.0.1:6379> set 1mars 0
OK
127.0.0.1:6379> incr 1mars
(integer) 1
127.0.0.1:6379> incr 1mars
(integer) 2
127.0.0.1:6379> incr 1mars
(integer) 3
127.0.0.1:6379> incr 1mars
(integer) 4
127.0.0.1:6379> incr 1mars
(integer) 5
127.0.0.1:6379> decr 1mars
(integer) 4
127.0.0.1:6379> decr 1mars
(integer) 3
127.0.0.1:6379> decr 1mars
(integer) 2
127.0.0.1:6379> set macle mavaleur
OK
127.0.0.1:6379> ttl macle
(integer) -1
127.0.0.1:6379> expire macle 120
(integer) 1
127.0.0.1:6379> ttl macle
(integer) 64
127.0.0.1:6379> RPUSH mesCours "BDA"
(integer) 1
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 2
127.0.0.1:6379> get mesCours
(error) WRONGTYPE Operation against a key holding the wrong kind of value
127.0.0.1:6379> LRANGE mesCours 0 -1
1) "BDA"
2) "Services Web"
127.0.0.1:6379> LRANGE mesCours 0 0
1) "BDA"
127.0.0.1:6379> LRANGE mesCours 1 1
1) "Services Web"
127.0.0.1:6379> LPOP mesCours
"BDA"
127.0.0.1:6379> LRANGE mesCours 0 -1
1) "Services Web"
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 2
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 3
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 4
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 5
127.0.0.1:6379> RPUSH mesCours "Services Web"
(integer) 6
127.0.0.1:6379> SADD utilisateurs "Augustin"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Ines"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Samir"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Marc"
(integer) 1
127.0.0.1:6379> SADD utilisateurs "Marc"
(integer) 0
127.0.0.1:6379>
```

```
(integer) 0
127.0.0.1:6379> SMEMBERS utilisateurs
1) "Augustin"
2) "Ines"
3) "Samir"
4) "Marc"
127.0.0.1:6379> SREM "Marc"
(error) ERR wrong number of arguments for 'srem' command
127.0.0.1:6379> SREM utilisateurs "Marc"
(integer) 1
127.0.0.1:6379> SREM utilisateurs "Marc"
(integer) 0
127.0.0.1:6379> SMEMEBERS utilisateurs
(error) ERR unknown command 'SMEMEBERS', with args beginning with: 'utilisateurs'

127.0.0.1:6379> SMEMBERS utilisateurs
1) "Augustin"
2) "Ines"
3) "Samir"
127.0.0.1:6379>
```

```
127.0.0.1:6379> SADD autresUtilisateurs
(error) ERR wrong number of arguments for 'sadd' command
127.0.0.1:6379> SADD autresUtilisateurs "Antoin"
(integer) 1
127.0.0.1:6379> SADD autresUtilisateurs "Philipe"
(integer) 1
127.0.0.1:6379> SUNION autresUtilisateurs utilisateurs
1) "Augustin"
2) "Ines"
3) "Antoin"
4) "Samir"
5) "Philipe"
127.0.0.1:6379>
```

VIDEO 2

```
127.0.0.1:6379> ZADD score4 19 "Augustin"
(integer) 1
127.0.0.1:6379> ZADD score4 19 "Ines"
(integer) 1
127.0.0.1:6379> ZADD score4 19 "Samir"
(integer) 1
127.0.0.1:6379> ZADD score4 19 "Philippe"
(integer) 1
127.0.0.1:6379> ZRANGE score4 0 -1
1) "Augustin"
2) "Ines"
3) "Philippe"
4) "Samir"
127.0.0.1:6379> ZRANGE score4 0 1
1) "Augustin"
2) "Ines"
127.0.0.1:6379> ZREVRANGE score4 0 -1
1) "Samir"
2) "Philippe"
3) "Ines"
4) "Augustin"
127.0.0.1:6379> ZRANK score4 "Augustin"
(integer) 0
127.0.0.1:6379>
```

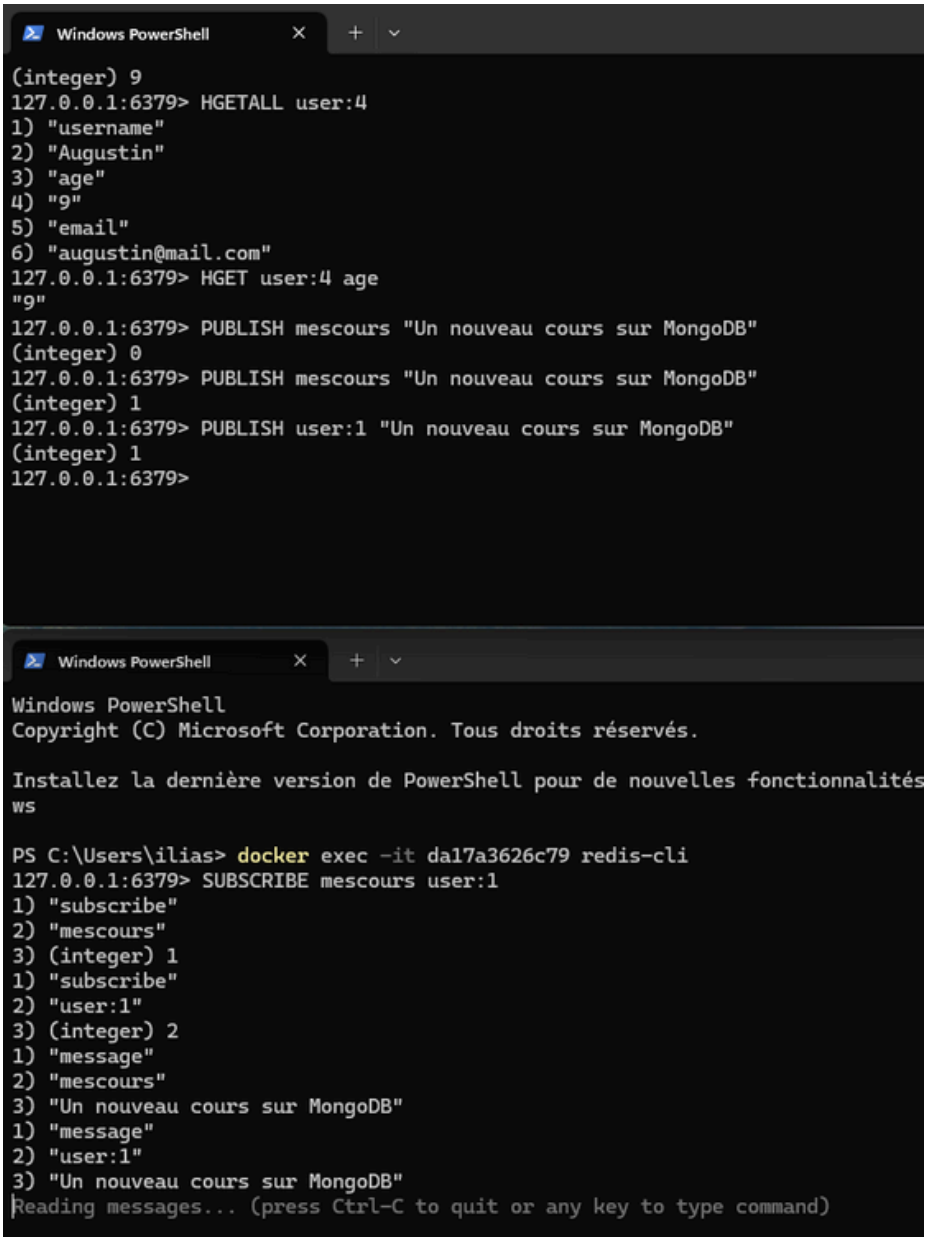
```
127.0.0.1:6379> HSET user:11 username "ilias"
(integer) 1
127.0.0.1:6379> HSET user:11 age 11
(integer) 1
127.0.0.1:6379> HSET user:11 email ilias@mail.com
(integer) 1
127.0.0.1:6379> HGETALL user:11
1) "username"
2) "ilias"
3) "age"
4) "11"
5) "email"
6) "ilias@mail.com"
127.0.0.1:6379> HMSET user:4 username "Augustin" age 5 email augustin@mail.com
OK
127.0.0.1:6379> HGETALL user:4
1) "username"
2) "Augustin"
3) "age"
4) "5"
5) "email"
6) "augustin@mail.com"
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@mail.com"
127.0.0.1:6379> HGET user:4 age
"9"
```

NOUVELLE COMMANDE EXEMPLE D'UTILISATION DE BITMAP

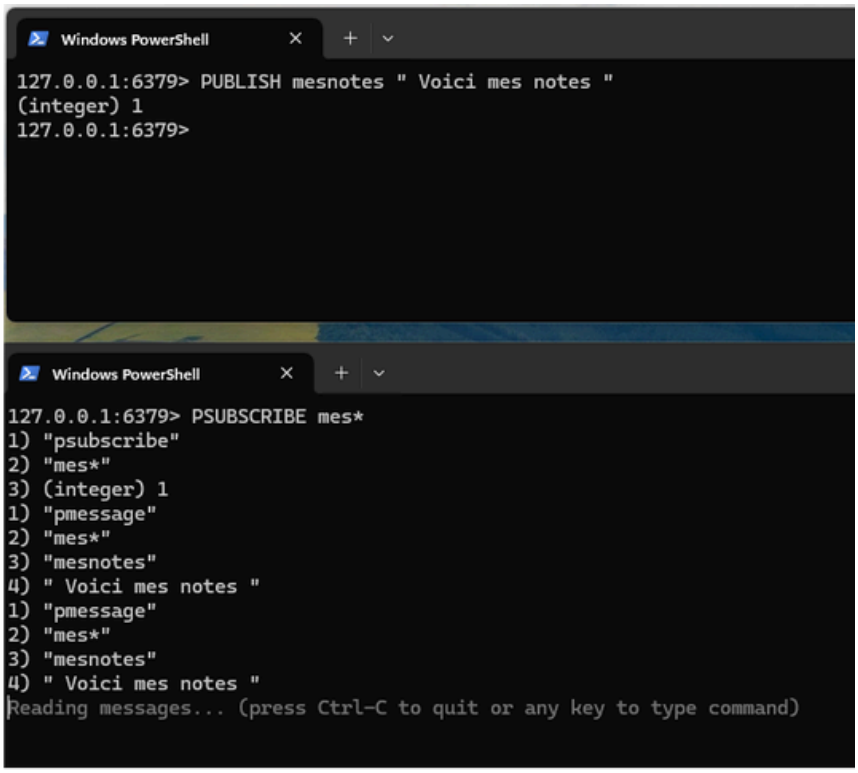
```
C:\Users\cloud>docker exec -it redis-server redis-cli
127.0.0.1:6379> setbit user:10 1 1
(integer) 0
127.0.0.1:6379> setbit user:10 5 1
(integer) 0
127.0.0.1:6379> setbit user:10 20 1
(integer) 0
127.0.0.1:6379> getbit user:10 5
(integer) 1
127.0.0.1:6379> bitcount user:10
(integer) 3
127.0.0.1:6379> setbit groupeA 1 1
(integer) 0
127.0.0.1:6379> setbit groupeA 3 1
(integer) 0
127.0.0.1:6379> setbit groupeB 3 1
(integer) 0
127.0.0.1:6379> setbit groupeB 4 1
(integer) 0
127.0.0.1:6379> bitop or unionGroups groupeA groupeB
(integer) 1
127.0.0.1:6379> bitop and interserctionGroups groupeA groupeB
(integer) 1
127.0.0.1:6379> bitop xor xorGroups groupeA groupeB
(integer) 1
127.0.0.1:6379> getbit unionGroups
(error) ERR wrong number of arguments for 'getbit' command
127.0.0.1:6379> get unionGroups
"X"
127.0.0.1:6379> get xorGroups
"H"
127.0.0.1:6379>
```

VIDEO 3

Souscription a un canal avec la commande SUBSCRIBE et la publication sur le canal avec la commande PUBLISH



Souscription a un canal où son nom commence avec “mes” et écoute decu



```
test> show dbs
admin                40.00 KiB
config               60.00 KiB
local                40.00 KiB
sample_airbnb        52.68 MiB
sample_analytics      8.95 MiB
sample_geospatial  1008.00 KiB
sample_guides        56.00 KiB
sample_mflix         94.77 MiB
sample_restaurants    6.20 MiB
sample_supplies    1000.00 KiB
sample_training      40.63 MiB
sample_weatherdata    2.42 MiB
test> use sample_mflix
switched to db sample_mflix
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. Everythi
ng is going well until the park's newest attraction--a genetically modified giant st
ealth killing 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/MV5BNzQ3OTY4NjAtNzM5OS00N2ZhLWJlOWU
tYzYwZjNmOWRiMzcyXkEyXkFqcGdeQXVyMTMxODk2OTU@._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 on the park genetically engineer a new breed of dinosaur. When everything
```

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

```
switched to db sample_mflix
sample_mflix> db.movies.find({ genres: "Comedy" })
[
  {
    _id: ObjectId('573a1390f29313caabcd4803'),
    plot: 'Cartoon figures announce, via comic strip balloons, that they will move -
 and move 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/MV5BYzg2NjNhNTctMjUxMi00ZWU4LWI3ZjY
tNTI0NTQxNThjZTk2XkEyXkFqcGdeQXVyNzg5OTk2OA@@._V1_SY1000_SX677_AL_.jpg',
    title: 'Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving Co
mics',
    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 dra
wing supplies, and the cartoonist must also overcome some mishaps caused by an assis
tant. Finally, the work is done, and everyone can see the resulting animated picture
.',
```

3. Afficher les sortis entre entre 2000 et 2005.

```
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
  },
```

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

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

5. Afficher les films sans champ rated .

```
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 Co
mics'
  },
  {
    _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'
```

Partie 2 – Agrégation

6. Afficher le nombre de films par année.

```
...   {$group: {_id: "$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.

```
Type "it" for more
...   {$unwind: "$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.

```
Type "it" for more
...   {$unwind: "$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.

```
sample_mflix> db.movies.aggregate([ { $unwind: "$directors" }, { $group: { _id: "$di
sample_mflix>
[ { _id: 'Woody Allen', total: 40 },
  { _id: 'Woody Allen', total: 40 },2 },
  { _id: 'Martin Scorsese', total: 32 },
  { _id: 'Takashi Miike', total: 31 },
  { _id: 'John Ford', total: 29 },: 29 }
  { _id: 'Steven Spielberg', total: 29 }
]
```

10. Afficher les films triés par note IMDb.

```
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: '' }
```

Partie 3 – Mises à jour

11. Ajouter un champ etat .

```
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: '' }
```

12. Incrémenter les votes IMDb de 100, par exemple

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

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

13. Supprimer le le champ poster

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

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
sample_mflix>
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 21349,
  modifiedCount: 18044,
  upsertedCount: 0
}
```

14. Modifier le réalisateur.

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

Partie 4 – Requêtes complexes

15. Afficher les films les mieux notés par décennie.

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

```
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: '' }
```

17. afficher les films avec plus de 2 genres.

```
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 Co
mics'
  },
  {
    _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'
  },
```

18. Afficher les films de Christopher Nolan.

```
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'
  },
```

Partie 5 – Indexation

19. Créer un index sur year

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

20. Vérifier les index existants.

```
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()).

```
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,
```

22. Supprimer l'index sur year .

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

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