

1. Création des répertoires de données

Dans le terminal CLIENT :

Trois répertoires de stockage ont été créés afin d'isoler les données du serveur de configuration et celles des deux shards.



```
Last login: Wed Dec 17 21:26:03 on ttys007
[hasnaelgarani@Air-de-hasna ~ % mkdir configsvrdb
[hasnaelgarani@Air-de-hasna ~ % mkdir serv1
[hasnaelgarani@Air-de-hasna ~ % mkdir serv2
hasnaelgarani@Air-de-hasna ~ % ]
```

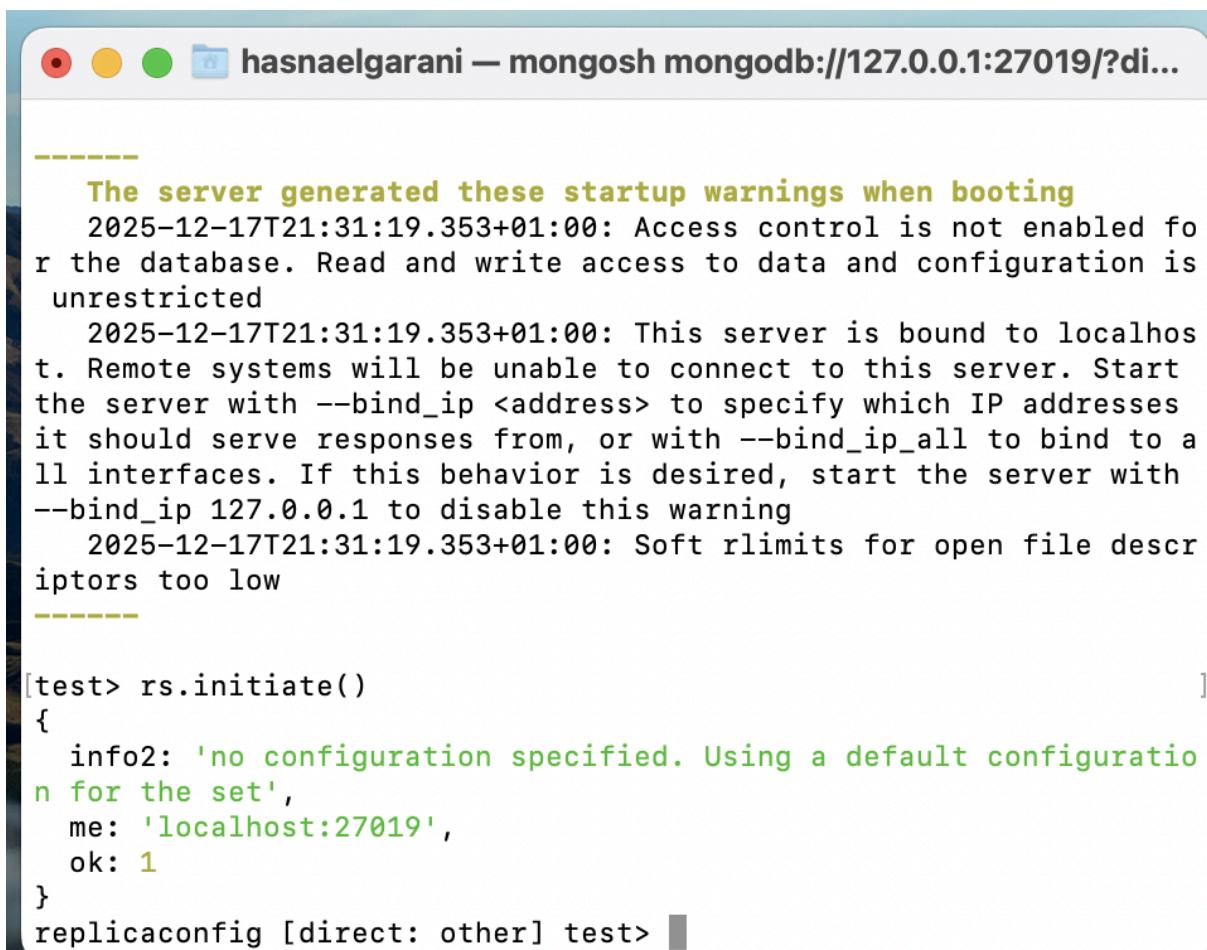
2. Mise en place du Config Server

Terminal : **CONFIG_SVR**

```
[hasnaelgarani@Air-de-hasna ~ % mongod --configsvr --replSet replic
aconfig --dbpath configsvrdb --port 27019
{"t": {"$date": "2025-12-17T21:31:18.347+01:00"}, "s": "I", "c": "NETW
ORK", "id": 4915701, "ctx": "thread1", "msg": "Initialized wire speci
fication", "attr": {"spec": {"incomingExternalClient": {"minWireVersio
n": 0, "maxWireVersion": 21}, "incomingInternalClient": {"minWireVersio
n": 0, "maxWireVersion": 21}, "outgoing": {"minWireVersion": 6, "maxWireV
ersion": 21}, "isInternalClient": true}}}
{"t": {"$date": "2025-12-17T21:31:18.370+01:00"}, "s": "I", "c": "CONT
ROL", "id": 23285, "ctx": "thread1", "msg": "Automatically disablin
g TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols
'none'"}
{"t": {"$date": "2025-12-17T21:31:18.371+01:00"}, "s": "I", "c": "NETW
ORK", "id": 4648602, "ctx": "thread1", "msg": "Implicit TCP FastOpen
in use."}
{"t": {"$date": "2025-12-17T21:31:18.376+01:00"}, "s": "I", "c": "REPL
", "id": 5123008, "ctx": "thread1", "msg": "Successfully registered
PrimaryOnlyService", "attr": {"service": "ReshardingCoordinatorServ
ice", "namespace": "config.reshardingOperations"}}
{"t": {"$date": "2025-12-17T21:31:18.376+01:00"}, "s": "I", "c": "REPL
", "id": 5123008, "ctx": "thread1", "msg": "Successfully registered
PrimaryOnlyService", "attr": {"service": "ConfigsvrCoordinatorServ
ice", "namespace": "config.configsvr_coordinators"}}
```

Dans le terminal CLIENT :

Un serveur de configuration a été démarré sur le port 27019. Celui-ci est intégré dans un replica set nommé replicaconfig, indispensable au bon fonctionnement du sharding, car il stocke les métadonnées de partitionnement du cluster.



```
hasnaelgarani — mongosh mongodb://127.0.0.1:27019/?di...
```

```
-----  
The server generated these startup warnings when booting  
2025-12-17T21:31:19.353+01:00: Access control is not enabled fo  
r the database. Read and write access to data and configuration is  
unrestricted  
2025-12-17T21:31:19.353+01:00: This server is bound to localhos  
t. Remote systems will be unable to connect to this server. Start  
the server with --bind_ip <address> to specify which IP addresses  
it should serve responses from, or with --bind_ip_all to bind to a  
ll interfaces. If this behavior is desired, start the server with  
--bind_ip 127.0.0.1 to disable this warning  
2025-12-17T21:31:19.353+01:00: Soft rlimits for open file descr  
iptors too low  
-----  
[test> rs.initiate()  
{  
    info2: 'no configuration specified. Using a default configuratio  
n for the set',  
    me: 'localhost:27019',  
    ok: 1  
}  
replicaconfig [direct: other] test> ]
```

3. Démarrage du routeur mongos

Terminal : MONGOS

Le routeur mongos a été lancé. Il joue le rôle d'intermédiaire entre les clients et les shards, en redirigeant les requêtes vers les serveurs appropriés en fonction de la clé de sharding.

```
[hasnaelgarani@Air-de-hasna ~ % mongos --configdb replicaconfig/loc
calhost:27019
{"t":{"$date":"2025-12-17T20:36:03.363Z"},"s":"W", "c":"SHARDING
", "id":24132, "ctx":"thread1", "msg":"Running a sharded cluster
with fewer than 3 config servers should only be done for testing
purposes and is not recommended for production."}
{"t":{"$date":"2025-12-17T21:36:03.365+01:00"},"s":"I", "c":"CON
TROL", "id":23285, "ctx":"thread1", "msg":"Automatically disabl
ing TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtoco
ls 'none'"}
 {"t":{"$date":"2025-12-17T21:36:03.365+01:00"},"s":"I", "c":"NET
WORK", "id":4915701, "ctx":"thread1", "msg":"Initialized wire spe
cification", "attr":{"spec":{"incomingExternalClient":{"minWireVer
sion":0,"maxWireVersion":21}, "incomingInternalClient":{"minWireVe
rsion":0,"maxWireVersion":21}, "outgoing":{"minWireVersion":21,"ma
xWireVersion":21}, "isInternalClient":true}}}
 {"t":{"$date":"2025-12-17T21:36:03.370+01:00"},"s":"I", "c":"NET
WORK", "id":4648602, "ctx":"thread1", "msg":"Implicit TCP FastOpe
n in use."}
 {"t":{"$date":"2025-12-17T21:36:03.370+01:00"},"s":"I", "c":"HEA
LTH", "id":5936503, "ctx":"thread1", "msg":"Fault manager change
d state ", "attr":{"state":"StartupCheck"}}
 {"t":{"$date":"2025-12-17T21:36:03.370+01:00"},"s":"W", "c":"CON
```

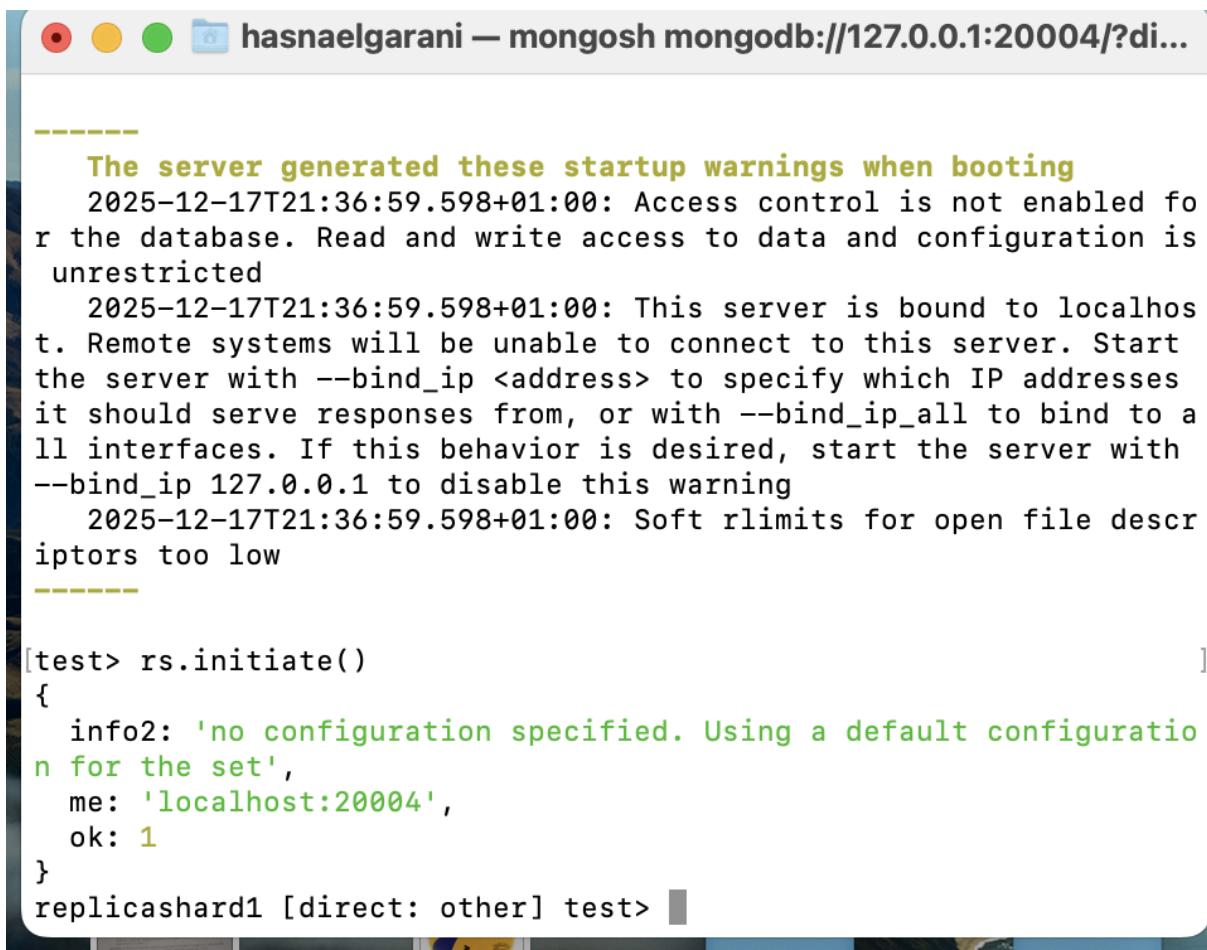
4. Mise en place des shards

♦ Shard 1

Terminal : SHARD1

```
[hasnaelgarani@Air-de-hasna ~ % mongod --replSet replicashard1  
--dbpath serv1 --shardsvr --port 20004  
{"t": {"$date": "2025-12-17T21:36:58.636+01:00"}, "s": "I", "c":  
"CONTROL", "id": 23285, "ctx": "thread1", "msg": "Automaticall  
y disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDis  
abledProtocols 'none'"}  
{"t": {"$date": "2025-12-17T21:36:58.658+01:00"}, "s": "I", "c":  
"NETWORK", "id": 4915701, "ctx": "thread1", "msg": "Initialized  
wire specification", "attr": {"spec": {"incomingExternalClient":  
{"minWireVersion": 0, "maxWireVersion": 21}, "incomingInternalCli  
ent": {"minWireVersion": 0, "maxWireVersion": 21}, "outgoing": {"mi  
nWireVersion": 6, "maxWireVersion": 21}, "isInternalClient": true}  
}  
{"t": {"$date": "2025-12-17T21:36:58.659+01:00"}, "s": "I", "c":  
"NETWORK", "id": 4648602, "ctx": "thread1", "msg": "Implicit TCP  
FastOpen in use."}  
{"t": {"$date": "2025-12-17T21:36:58.662+01:00"}, "s": "I", "c":  
"REPL", "id": 5123008, "ctx": "thread1", "msg": "Successfully  
registered PrimaryOnlyService", "attr": {"service": "RenameColl  
ectionParticipantService", "namespace": "config.localRenamePart  
icipants"}}  
{"t": {"$date": "2025-12-17T21:36:58.662+01:00"}, "s": "I", "c":
```

Dans CLIENT :



```
hasnaelgarani — mongosh mongodb://127.0.0.1:20004/?di...
```

```
-----  
The server generated these startup warnings when booting  
2025-12-17T21:36:59.598+01:00: Access control is not enabled fo  
r the database. Read and write access to data and configuration is  
unrestricted  
2025-12-17T21:36:59.598+01:00: This server is bound to localhos  
t. Remote systems will be unable to connect to this server. Start  
the server with --bind_ip <address> to specify which IP addresses  
it should serve responses from, or with --bind_ip_all to bind to a  
ll interfaces. If this behavior is desired, start the server with  
--bind_ip 127.0.0.1 to disable this warning  
2025-12-17T21:36:59.598+01:00: Soft rlimits for open file descr  
iptors too low  
-----  
[test> rs.initiate()  
{  
  info2: 'no configuration specified. Using a default configuratio  
n for the set',  
  me: 'localhost:20004',  
  ok: 1  
}  
replicashard1 [direct: other] test>
```

◆ Shard 2

Terminal : SHARD2

```
[hasnaelgarani@Air-de-hasna ~ % mongod --repSet replicashard2 --d]
bpath serv2 --shardsvr --port 20005
{"t":{"$date":"2025-12-17T21:38:33.421+01:00"}, "s": "I", "c": "NET
WORK", "id": 4915701, "ctx": "thread1", "msg": "Initialized wire spe
cification", "attr": {"spec": {"incomingExternalClient": {"minWireVer
sion": 0, "maxWireVersion": 21}, "incomingInternalClient": {"minWireVe
rsion": 0, "maxWireVersion": 21}, "outgoing": {"minWireVersion": 6, "max
WireVersion": 21}, "isInternalClient": true}}}
{"t":{"$date":"2025-12-17T21:38:33.423+01:00"}, "s": "I", "c": "CON
TROL", "id": 23285, "ctx": "thread1", "msg": "Automatically disabl
ing TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtoco
ls 'none'"}
{"t":{"$date":"2025-12-17T21:38:33.423+01:00"}, "s": "I", "c": "NET
WORK", "id": 4648602, "ctx": "thread1", "msg": "Implicit TCP FastOpe
n in use."}
{"t":{"$date":"2025-12-17T21:38:33.424+01:00"}, "s": "I", "c": "REP
L", "id": 5123008, "ctx": "thread1", "msg": "Successfully registe
red PrimaryOnlyService", "attr": {"service": "RenameCollectionPartic
ipantService", "namespace": "config.localRenameParticipants"}}
{"t":{"$date":"2025-12-17T21:38:33.424+01:00"}, "s": "I", "c": "REP
L", "id": 5123008, "ctx": "thread1", "msg": "Successfully registe
red PrimaryOnlyService", "attr": {"service": "ShardingDDLCoordinator
", "namespace": "config.system.sharding_ddl_coordinators"}}
{"t":{"$date":"2025-12-17T21:38:33.424+01:00"}, "s": "I", "c": "REP
```

Dans CLIENT :

```
hasnaelgarani — mongosh mongodb://127.0.0.1:20005/?di...
```

```
-----  
The server generated these startup warnings when booting  
2025-12-17T21:38:34.389+01:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted  
2025-12-17T21:38:34.390+01:00: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning  
2025-12-17T21:38:34.390+01:00: Soft rlimits for open file descriptors too low  
-----  
[test> rs.initiate()  
{  
    info2: 'no configuration specified. Using a default configuration for the set',  
    me: 'localhost:20005',  
    ok: 1  
}  
replicashard2 [direct: other] test>
```

Deux shards ont été déployés, chacun configuré comme un replica set. Cette approche permet d'assurer la tolérance aux pannes et la montée en charge du cluster.

5. Ajout des shards au cluster

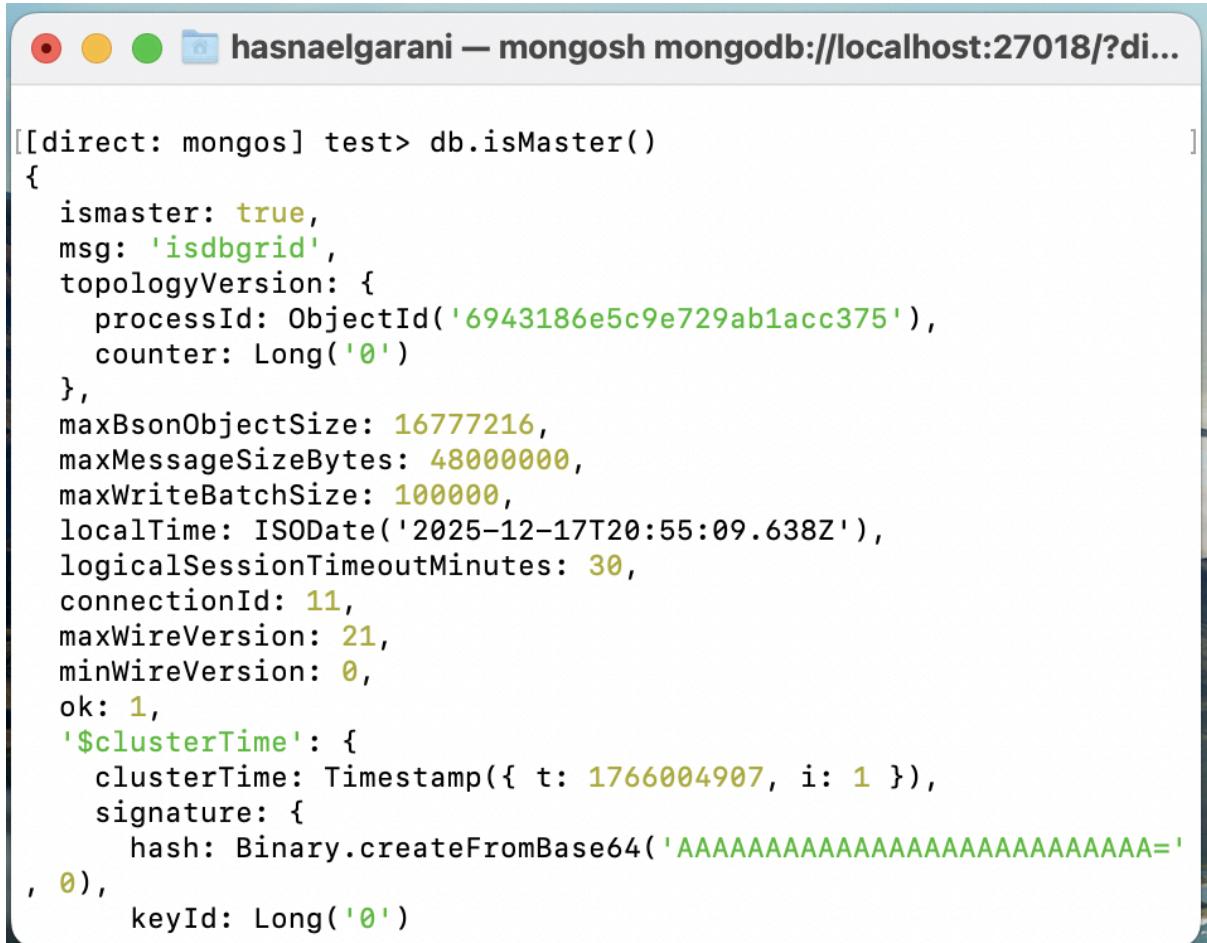
Terminal : CLIENT (connecté à mongos)

```
hasnaelgarani — mongosh mongodb://localhost:27018/?di...
hasnaelgarani@Air-de-hasna ~ % mongosh --host localhost --port 270
18

Current Mongosh Log ID: 6943189f83c2b56eeec4c5f0
Connecting to:      mongodb://localhost:27018/?directConnectio
n=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.9
Using MongoDB:     7.0.26
Using Mongosh:    2.5.9
mongosh 2.5.10 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

-----
The server generated these startup warnings when booting
2025-12-17T21:54:06.236+01:00: Access control is not enabled fo
r the database. Read and write access to data and configuration is
unrestricted
2025-12-17T21:54:06.236+01:00: This server is bound to localhos
t. Remote systems will be unable to connect to this server. Start
the server with --bind_ip <address> to specify which IP addresses
it should serve responses from, or with --bind_ip_all to bind to a
ll interfaces. If this behavior is desired, start the server with
--bind_ip 127.0.0.1 to disable this warning
```



```
[direct: mongos] test> db.isMaster()
{
  ismaster: true,
  msg: 'isdbgrid',
  topologyVersion: {
    processId: ObjectId('6943186e5c9e729ab1acc375'),
    counter: Long('0')
  },
  maxBsonObjectSize: 16777216,
  maxMessageSizeBytes: 48000000,
  maxWriteBatchSize: 100000,
  localTime: ISODate('2025-12-17T20:55:09.638Z'),
  logicalSessionTimeoutMinutes: 30,
  connectionId: 11,
  maxWireVersion: 21,
  minWireVersion: 0,
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1766004907, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA='),
      0,
      keyId: Long('0')
    }
  }
}
```

Les deux shards ont été ajoutés dynamiquement au cluster MongoDB à l'aide du routeur mongos.

```

}
[direct: mongos] test> sh.addShard("replicashard1/localhost:20004")
)
{
  shardAdded: 'replicashard1',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1766005073, i: 5 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA='),
      0,
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1766005073, i: 5 })
}
[direct: mongos] test>

```

```

}
[direct: mongos] test> sh.addShard("replicashard2/localhost:20005")
)
{
  shardAdded: 'replicashard2',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1766005107, i: 13 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA='),
      0,
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1766005107, i: 3 })
}
[direct: mongos] test>

```

6. Activation du sharding sur la base

Le sharding n'étant pas activé par défaut, il a été explicitement activé sur la base de données mabasefilms.

```
[  
]  
[direct: mongos] test> sh.enableSharding("mabasefilms")  
[...]  
{  
    ok: 1,  
    '$clusterTime': {  
        clusterTime: Timestamp({ t: 1766005196, i: 9 }),  
        signature: {  
            hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA=')  
, 0),  
            keyId: Long('0')  
        }  
    },  
    operationTime: Timestamp({ t: 1766005196, i: 3 })  
}  
[direct: mongos] test> █
```

7. Sharding de la collection

La collection films a été shardée en utilisant le champ titre comme clé de partitionnement. Cette clé permet de répartir les documents entre les shards selon des plages de valeurs.

```
[  
]  
[direct: mongos] test> sh.shardCollection("mabasefilms.films", { "titre": 1 })  
[...]  
{  
    collectionsharded: 'mabasefilms.films',  
    ok: 1,  
    '$clusterTime': {  
        clusterTime: Timestamp({ t: 1766005256, i: 8 }),  
        signature: {  
            hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAA=')  
, 0),  
            keyId: Long('0')  
        }  
    },  
    operationTime: Timestamp({ t: 1766005256, i: 8 })  
}  
[direct: mongos] test> █
```

8. Insertion des données et observation

Lors de l'insertion massive des films, MongoDB a automatiquement découpé les données en chunks et les a redistribués entre les shards afin d'équilibrer la charge, grâce au balancer.

```
directory
● (monenv) hasnaelgarani@Air-de-hasna Downloads % python monappunparun.py
Insertion film par film en cours...
Insertion terminée. Temps total : 83.43 secondes.
○ (monenv) hasnaelgarani@Air-de-hasna Downloads % █
L 1, col 1 Espaces : 4 UTF-8 L
```

→ La commande sh.status() :

Cette commande permet de vérifier l'état général du cluster sharded, en affichant les shards, les chunks et l'activité du balancer.

```

hasnaelgarani — mongosh mongodb://localhost:27018/?directC
}
},
operationTime: Timestamp({ t: 1766008547, i: 1 })
}
[[direct: mongos] mabasefilms> sh.status()
shardingVersion
{ _id: 1, clusterId: ObjectId('6943140386e0ee5a54b7358c') }
---
shards
[
{
  _id: 'replicashard1',
  host: 'replicashard1/localhost:20004',
  state: 1,
  topologyTime: Timestamp({ t: 1766005073, i: 2 })
},
{
  _id: 'replicashard2',
  host: 'replicashard2/localhost:20005',
  state: 1,
  topologyTime: Timestamp({ t: 1766005107, i: 1 })
}
]
---
active mongoses
[ { '7.0.26': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Failed balancer rounds in last 5 attempts': 0,
  'Currently running': 'no',
  'Migration Results for the last 24 hours': 'No recent migrations'
}
---
shardedDataDistribution
[
  {
    ns: 'config.system.sessions',
    shards: [
      {
        shardName: 'replicashard1',
        numOrphanedDocs: 0,
    }
  }
]

```

→ La commande db.printShardingStatus():

Cette commande fournit des informations détaillées sur le sharding de chaque collection, la clé de sharding utilisée, et la répartition des chunks entre les shards.

```
[hasnaelgarani — mongosh mongodb://localhost:27018/?directC
]
[direct: mongos] mabasefilms> db.printShardingStatus()
shardingVersion
{ _id: 1, clusterId: ObjectId('6943140386e0ee5a54b7358c') }
---
shards
[
  {
    _id: 'replicashard1',
    host: 'replicashard1/localhost:20004',
    state: 1,
    topologyTime: Timestamp({ t: 1766005073, i: 2 })
  },
  {
    _id: 'replicashard2',
    host: 'replicashard2/localhost:20005',
    state: 1,
    topologyTime: Timestamp({ t: 1766005107, i: 1 })
  }
]
---
active mongoses
[ { '7.0.26': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Currently running': 'no',
  'Failed balancer rounds in last 5 attempts': 0,
  'Migration Results for the last 24 hours': 'No recent migrations'
}
---
shardedDataDistribution
[
  {
    ns: 'config.system.sessions',
    shards: [
      {
        shardName: 'replicashard1',
        numOrphanedDocs: 0,
        numOwnedDocuments: 11,
        ownedSizeBytes: 1089,
        orphanedSizeBytes: 0
      }
    ]
  }
]
```

→ La commande db.stats():

Cette commande affiche les statistiques globales de la base de données sélectionnée, notamment le nombre de documents, la taille des données et l'espace utilisé par les indexés.

```
[direct: mongos] mabasefilms> db.stats()
{
  raw: {
    'replicashard1/localhost:20004': {
      db: 'mabasefilms',
      collections: Long('0'),
      views: Long('0'),
      objects: Long('0'),
      avgObjSize: 0,
      dataSize: 0,
      storageSize: 0,
      indexes: Long('0'),
      indexSize: 0,
      totalSize: 0,
      scaleFactor: Long('1'),
      fsUsedSize: 0,
      fsTotalSize: 0,
      ok: 1
    },
    'replicashard2/localhost:20005': {
      db: 'mabasefilms',
      collections: Long('1'),
      views: Long('0'),
      objects: Long('112482'),
      avgObjSize: 129.99284329937234,
      dataSize: 14621855,
      storageSize: 4550656,
      indexes: Long('2'),
      indexSize: 5324800,
      totalSize: 9875456,
      scaleFactor: Long('1'),
      fsUsedSize: 220412985344,
      fsTotalSize: 245107195904,
      ok: 1
    }
  },
  db: 'mabasefilms',
  collections: 1,
  views: 0,
  objects: 112482,
  avgObjSize: 129.99284329937234,
  dataSize: 14621855,
  storageSize: 4550656,
  totalSize: 9875456,
  indexes: 2,
  indexSize: 5324800,
```

1. Qu'est-ce que le sharding dans MongoDB et pourquoi est-il utilisé ?

Le sharding est le partitionnement horizontal des données pour répartir la charge et gérer de grands volumes de données de manière scalable.

2. Quelle est la différence entre le sharding et la réPLICATION dans MongoDB ?

La réPLICATION assure la redondance et la haute disponibilité des données, tandis que le sharding distribue les données sur plusieurs serveurs pour l'évolutivité.

3. Quels sont les composants d'une architecture shardée (mongos, config servers, shards) ?

Les shards stockent les données, les config servers conservent la configuration du cluster, et les mongos route les requêtes vers les shards appropriés.

4. Quelles sont les responsabilités des config servers (CSRS) dans un cluster shardé ?

Ils stockent la métadonnée du cluster, la carte des chunks et la configuration du sharding.

5. Quel est le rôle du mongos router ?

Le mongos dirige les requêtes clients vers les shards corrects en utilisant les informations des config servers.

6. Comment MongoDB décide-t-il sur quel shard stocker un document ?

MongoDB utilise la clé de sharding pour déterminer le chunk correspondant et stocke le document sur le shard qui possède ce chunk.

7. Qu'est-ce qu'une clé de sharding et pourquoi est-elle essentielle ?

C'est un ou plusieurs champs définissant la distribution des documents entre shards, essentielle pour équilibrer la charge et optimiser les requêtes.

8. Quels sont les critères de choix d'une bonne clé de sharding ?

Cardinalité élevée, uniformité de distribution, stabilité et pertinence par rapport aux requêtes fréquentes.

9. Qu'est-ce qu'un chunk dans MongoDB ?

Un chunk est un sous-ensemble contigu de documents dans une collection, basé sur la clé de sharding.

10. Comment fonctionne le splitting des chunks ?

MongoDB divise automatiquement un chunk lorsqu'il dépasse une taille limite (par défaut 64 Mo) pour équilibrer les données.

11. Que fait le balancer dans un cluster shardé ?

Il redistribue les chunks entre shards pour équilibrer la charge et optimiser l'espace de stockage.

12. Quand et comment le balancer déplace-t-il des chunks ?

Lorsqu'un shard devient surchargé, le balancer migre des chunks vers des shards moins chargés, automatiquement en arrière-plan.

13. Qu'est-ce qu'un hot shard et comment l'éviter ?

Un shard qui reçoit trop de trafic ou de données ; on l'évite en choisissant une clé de sharding bien distribuée.

14. Quels problèmes une clé de sharding monotone peut-elle engendrer ?

Elle peut créer un hotspot, concentrer l'écriture sur un seul shard et réduire les performances.

15. Comment activer le sharding sur une base de données et sur une collection ?

```
sh.enableSharding("mabasefilms")
sh.shardCollection("mabasefilms.films", { "titre": 1 })
```

16. Comment ajouter un nouveau shard à un cluster MongoDB ?

```
sh.addShard("replicaName/host:port")
```

17. Comment vérifier l'état du cluster shardé (commandes usuelles) ?

```
sh.status(), db.printShardingStatus(), db.stats()
```

18. Dans quels cas faut-il envisager d'utiliser un hashed sharding key ?

Lorsque la distribution uniforme des documents est prioritaire, indépendamment de la valeur logique des données.

19. Dans quels cas faut-il privilégier un ranged sharding key ?

Lorsque les requêtes sont souvent basées sur des plages de valeurs et que l'ordre des données est important.

20. Qu'est-ce que le zone sharding et quel est son intérêt ?

C'est l'assignation de chunks à des zones spécifiques pour contrôler la localisation des données et optimiser la géodistribution ou la conformité.

21. Comment MongoDB gère-t-il les requêtes multi-shards ?

Le mongos divise la requête et interroge tous les shards concernés, puis combine les résultats avant de les renvoyer au client.

22. Comment optimiser les performances de requêtes dans un environnement shardé ?

Utiliser des clés de sharding adaptées, créer des indexes sur la clé de sharding et limiter les requêtes aux shards ciblés.

23. Que se passe-t-il lorsqu'un shard devient indisponible ?

Les données sur ce shard deviennent inaccessibles, mais les autres shards restent opérationnels ; la réPLICATION peut assurer la continuité si configurée.

24. Comment migrer une collection existante vers un schéma shardé ?

Créer la collection shardée avec sh.shardCollection() et laisser MongoDB redistribuer les documents selon la clé de sharding.

25. Quels outils ou métriques utiliser pour diagnostiquer les problèmes de sharding ?

sh.status(), db.printShardingStatus(), db.currentOp(), logs MongoDB, et monitoring via MongoDB.