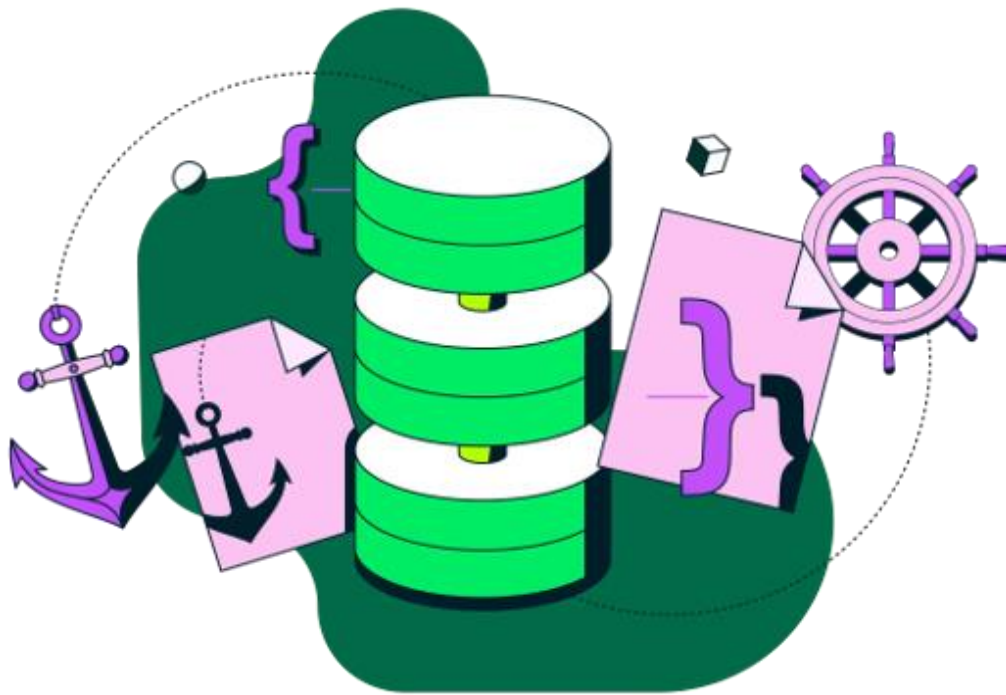


# La Réplication des Données dans MongoDB



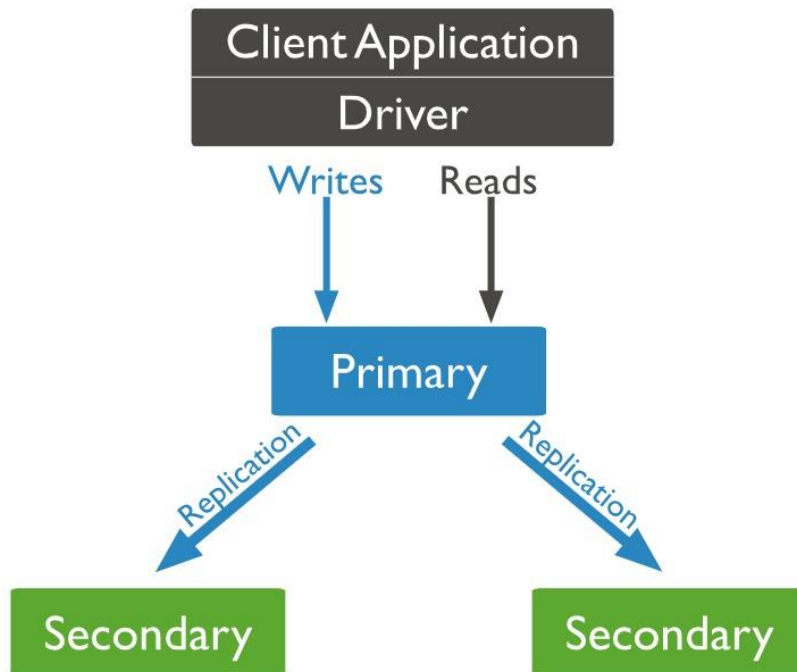
Présenté par:

SAHIB ILYASS  
EL HOUMA NOUHAILA  
NAHHAS KHANSAA  
WAHBI AYA  
SASSAOUI ABDELBASSET  
ZAYD WISSAL  
IGHAOUS ANASS

Encadré par:

Pr. HANOUNE MOSTAFA  
Mr. AZAMI JIHAD

Dans cet atelier, nous allons mettre en place un cluster MongoDB composé de 3 nœuds configurés en Replica Set.



## 1. Préparation de l'Environnement

Chaque nœud doit avoir son propre dossier pour stocker les données.

Pour créer le dossier : `mkdir mogo-replica`

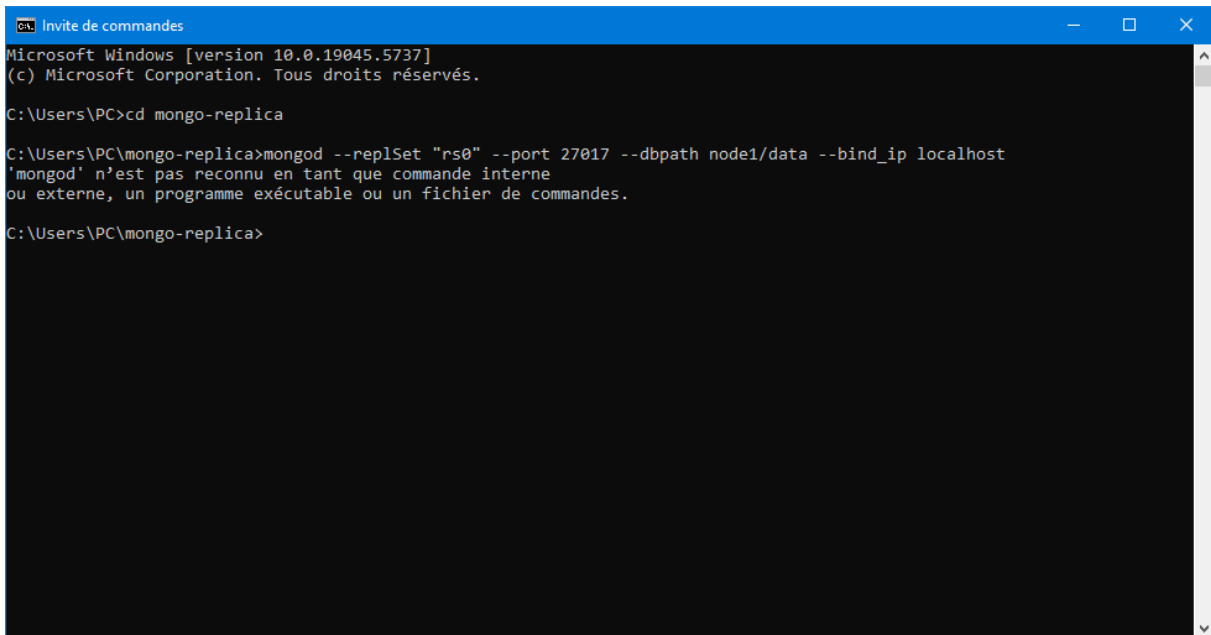
Pour créer les nœuds : `mkdir node1\data`

`mkdir node2\data`

`mkdir node3\data`

```
C:\Users\user>mkdir mongo-replica  
C:\Users\user>cd mongo-replica  
C:\Users\user\mongo-replica>mkdir node1\data  
C:\Users\user\mongo-replica>mkdir node2\data  
C:\Users\user\mongo-replica>mkdir node3\data
```

**Si vous rencontrez ce problème : 'mongod' n'est pas reconnu en tant que commande interne.**



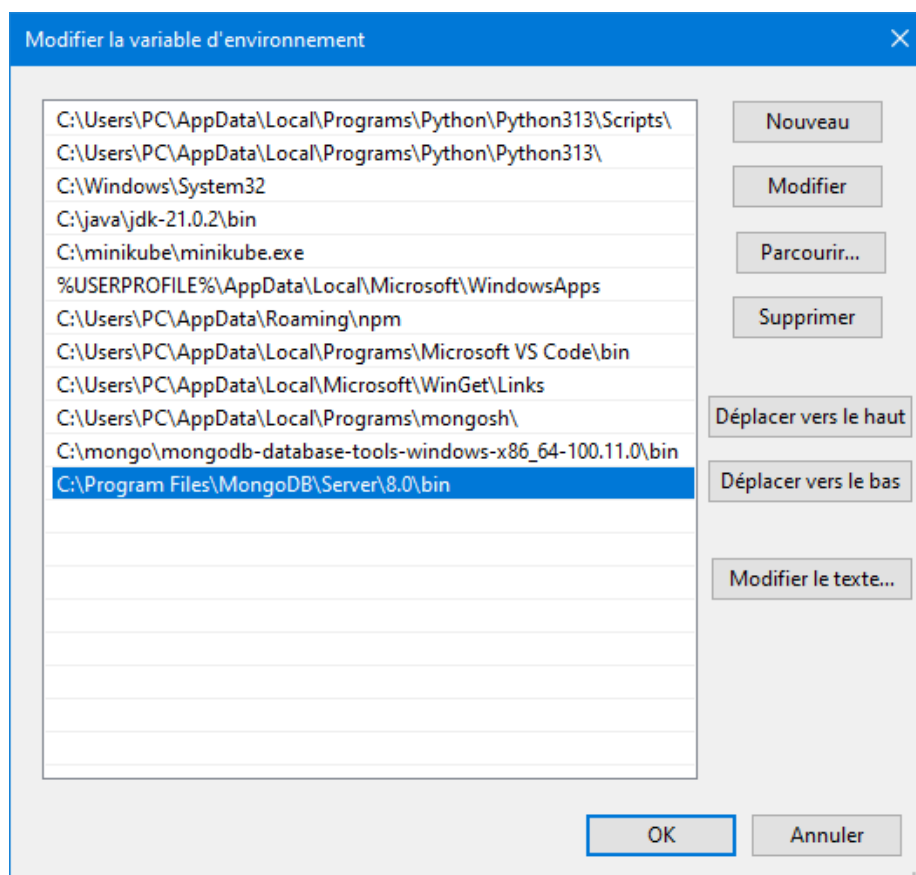
```
Invite de commandes
Microsoft Windows [version 10.0.19045.5737]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\PC>cd mongo-replica

C:\Users\PC\mongo-replica>mongod --replSet "rs0" --port 27017 --dbpath node1/data --bind_ip localhost
'mongod' n'est pas reconnu en tant que commande interne
ou externe, un programme exécutable ou un fichier de commandes.

C:\Users\PC\mongo-replica>
```

Vous devez ajouter le chemin du dossier bin, généralement : « C:\Program Files\MongoDB\Server\8.0\bin », à la variable d'environnement PATH. Puis, fermez l'invite de commande et réessayez.



## 2. Configuration des Nœuds

Démarrer 3 instances MongoDB avec des ports différents, chaque commande sur une invite de commandes séparées en même dossier "mongo-replica".

### a. Démarrer le Nœud 1 (Port 27017) :

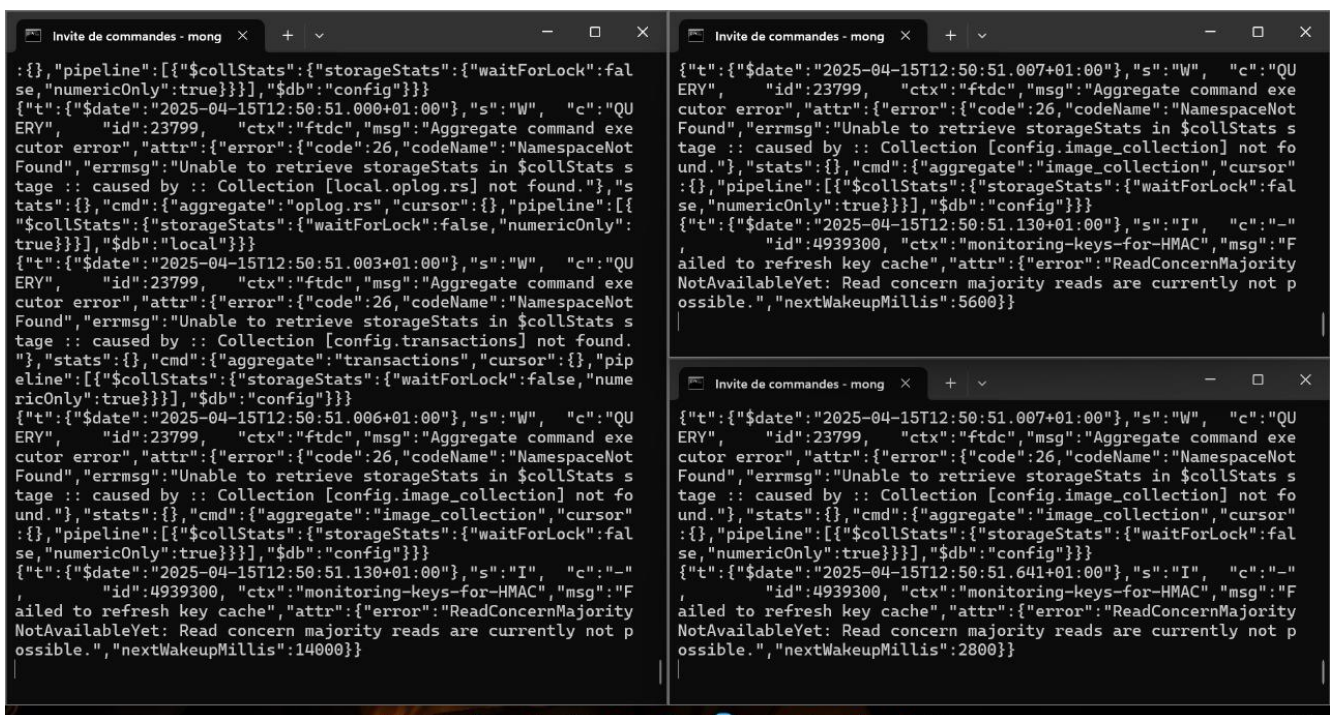
Avec cette commande "mongod --replSet "rs0" --port 27017 --dbpath node1/data --bind\_ip localhost".

### b. Démarrer le Nœud 2 (Port 27018) :

Avec cette commande "mongod --replSet "rs0" --port 27018 --dbpath node2/data --bind\_ip localhost".

### c. Démarrer le Nœud 3 (Port 27019) :

Avec cette commande "mongod --replSet "rs0" --port 27019 --dbpath node3/data --bind\_ip localhost".



```

:{"pipeline":[{"$collStats":{"storageStats":{"waitForLock":false,"numericOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.000+01:00"},"s":"W", "c":"QU
ERY", "id":23799, "ctx":"ftdc","msg":"Aggregate command exe
cutor error","attr":{"error":{"code":26,"codeName":"NamespaceNot
Found"},"errmsg":"Unable to retrieve storageStats in $collStats s
tage :: caused by :: Collection [config.image_collection] not fo
und."},"stats":{"cmd":{"aggregate":"image_collection"},"cursor"
:{"pipeline":[{"$collStats":{"storageStats":{"waitForLock":fal
se,"numericOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.003+01:00"},"s":"W", "c":"QU
ERY", "id":23799, "ctx":"ftdc","msg":"Aggregate command exe
cutor error","attr":{"error":{"code":26,"codeName":"NamespaceNot
Found"},"errmsg":"Unable to retrieve storageStats in $collStats s
tage :: caused by :: Collection [config.transactions] not found.
"},"stats":{"cmd":{"aggregate":"transactions"},"cursor":{"pip
eline":[{"$collStats":{"storageStats":{"waitForLock":false,"nume
ricOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.006+01:00"},"s":"W", "c":"QU
ERY", "id":23799, "ctx":"ftdc","msg":"Aggregate command exe
cutor error","attr":{"error":{"code":26,"codeName":"NamespaceNot
Found"},"errmsg":"Unable to retrieve storageStats in $collStats s
tage :: caused by :: Collection [config.image_collection] not fo
und."},"stats":{"cmd":{"aggregate":"image_collection"},"cursor"
:{"pipeline":[{"$collStats":{"storageStats":{"waitForLock":fal
se,"numericOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.130+01:00"},"s":"I", "c":"-
", "id":4939300, "ctx":"monitoring-keys-for-HMAC","msg":"F
ailed to refresh key cache","attr":{"error":"ReadConcernMajority
NotAvailableYet: Read concern majority reads are currently not p
ossible."},"nextWakeupMillis":14000}

{"t":{"$date":"2025-04-15T12:50:51.007+01:00"},"s":"W", "c":"QU
ERY", "id":23799, "ctx":"ftdc","msg":"Aggregate command exe
cutor error","attr":{"error":{"code":26,"codeName":"NamespaceNot
Found"},"errmsg":"Unable to retrieve storageStats in $collStats s
tage :: caused by :: Collection [config.image_collection] not fo
und."},"stats":{"cmd":{"aggregate":"image_collection"},"cursor"
:{"pipeline":[{"$collStats":{"storageStats":{"waitForLock":fal
se,"numericOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.130+01:00"},"s":"I", "c":"-
", "id":4939300, "ctx":"monitoring-keys-for-HMAC","msg":"F
ailed to refresh key cache","attr":{"error":"ReadConcernMajority
NotAvailableYet: Read concern majority reads are currently not p
ossible."},"nextWakeupMillis":14000}

{"t":{"$date":"2025-04-15T12:50:51.007+01:00"},"s":"W", "c":"QU
ERY", "id":23799, "ctx":"ftdc","msg":"Aggregate command exe
cutor error","attr":{"error":{"code":26,"codeName":"NamespaceNot
Found"},"errmsg":"Unable to retrieve storageStats in $collStats s
tage :: caused by :: Collection [config.image_collection] not fo
und."},"stats":{"cmd":{"aggregate":"image_collection"},"cursor"
:{"pipeline":[{"$collStats":{"storageStats":{"waitForLock":fal
se,"numericOnly":true}}},"$db":"config"}]}
{"t":{"$date":"2025-04-15T12:50:51.130+01:00"},"s":"I", "c":"-
", "id":4939300, "ctx":"monitoring-keys-for-HMAC","msg":"F
ailed to refresh key cache","attr":{"error":"ReadConcernMajority
NotAvailableYet: Read concern majority reads are currently not p
ossible."},"nextWakeupMillis":14000}
  
```

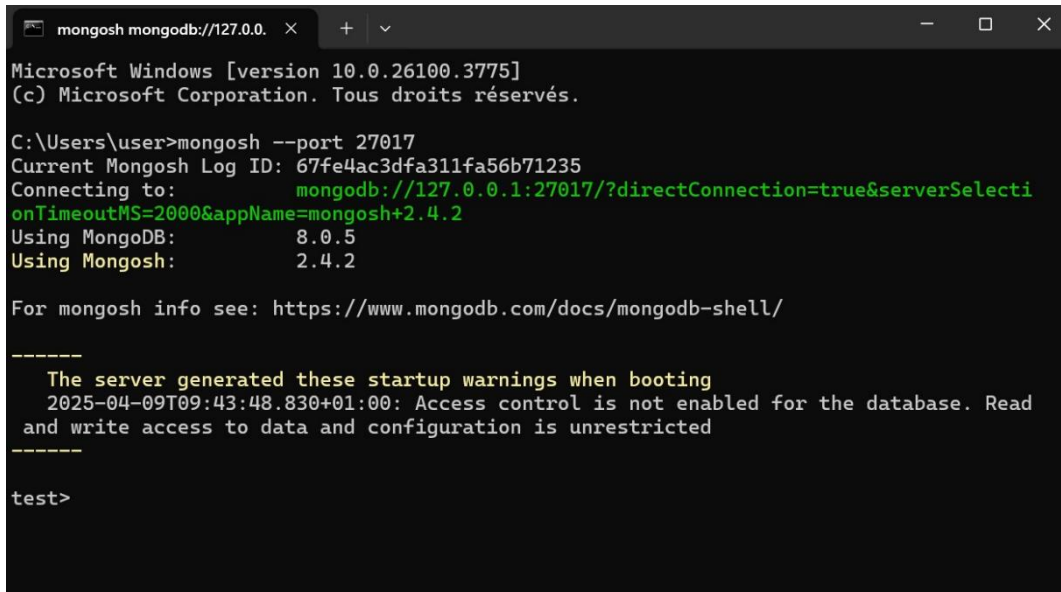
## 3. Initialisation du Replica Set:

On va lier les 3 nœuds en un cluster cohérent.

Pour cela, ouvrir une nouvelle invite de commande pour se connecter au serveur qui sera choisir comme serveur primaire.

## a. Connexion au Nœud 1:

Avec cette commande "mongosh --port 27017"



```
mongosh mongodb://127.0.0.1:27017
Microsoft Windows [version 10.0.26100.3775]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\user>mongosh --port 27017
Current Mongosh Log ID: 67fe4ac3dfa311fa56b71235
Connecting to:  mongodb://127.0.0.1:27017/?directConnection=true&serverSelecti
onTimeoutMS=2000&appName=mongosh+2.4.2
Using MongoDB: 8.0.5
Using Mongosh: 2.4.2

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

-----
The server generated these startup warnings when booting
2025-04-09T09:43:48.830+01:00: Access control is not enabled for the database. Read
and write access to data and configuration is unrestricted
-----

test>
```

## b. Configuration Initiale :

Définir l'identifiant du cluster (rs0) et l'adresse des membres. Voilà la commande :

```
"rs.initiate({
  _id: "rs0",
  members: [
    { _id: 0, host: "localhost:27017" },
    { _id: 1, host: "localhost:27018" },
    { _id: 2, host: "localhost:27019" }
  ]
});"
```

**Si vous rencontrez un problème, veuillez suivre les étapes ci-dessus :**

### 1. Activer la réplication :

Allez dans le fichier de config MongoDB (généralement situé ici : C:\Program Files\MongoDB\Server\8.0\bin\mongod.cfg) et ajoutez ou modifiez la section suivante :

```
replication:
  replSetName: "rs0"
```

2. Ouvrez le terminal en tant qu'administrateur, puis exécutez les commandes suivantes :

```
net stop MongoDB
```

```
net start MongoDB
```

3. Ensuite, ouvrez un autre terminal (CMD) et lancez les commandes suivantes :

```
mongosh --port 27017
```

```
rs.initiate({  
  _id: "rs0",  
  members: [  
    { _id: 0, host: "localhost:27017" },  
    { _id: 1, host: "localhost:27018" },  
    { _id: 2, host: "localhost:27019" }  
  ]  
});
```

```
test> rs.initiate({  
...   _id: "rs0",  
...   members: [  
...     { _id: 0, host: "localhost:27017" },  
...     { _id: 1, host: "localhost:27018" },  
...     { _id: 2, host: "localhost:27019" }  
...   ]  
... })  
...  
{  
  ok: 1,  
  '$clusterTime': {  
    clusterTime: Timestamp({ t: 1744718926, i: 1 }),  
    signature: {  
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),  
      keyId: Long('0')  
    }  
  },  
  operationTime: Timestamp({ t: 1744718926, i: 1 })  
}  
rs0 [direct: secondary] test> |
```



## c. Vérification :

Affiche le statut des nœuds (Primary/Secondary), avec cette commande "rs.status();"

```
rs0 [direct: secondary] test> rs.status()
{
  set: 'rs0',
  date: ISODate('2025-04-15T12:10:52.926Z'),
  myState: 1,
  term: Long('1'),
  syncSourceHost: '',
  syncSourceId: -1,
  heartbeatIntervalMillis: Long('2000'),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
  writableVotingMembersCount: 3,
  optimes: {
    lastCommittedOpTime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    lastCommittedWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    readConcernMajorityOpTime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    appliedOpTime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    durableOpTime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    writtenOpTime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    lastAppliedWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    lastDurableWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    lastWrittenWallTime: ISODate('2025-04-15T12:10:47.851Z')
  },
  lastStableRecoveryTimestamp: Timestamp({ t: 1744719037, i: 1 }),
  electionCandidateMetrics: {
    lastElectionReason: 'electionTimeout',
    lastElectionDate: ISODate('2025-04-15T12:08:57.528Z'),
    electionTerm: Long('1'),
    lastCommittedOpTimeAtElection: { ts: Timestamp({ t: 1744718926, i: 1 }), t: Long('1') },
    lastSeenWrittenOpTimeAtElection: { ts: Timestamp({ t: 1744718926, i: 1 }), t: Long('1') },
  },
}
```

```
members: [
  {
    _id: 0,
    name: 'localhost:27017',
    health: 1,
    state: 1,
    stateStr: 'PRIMARY',
    uptime: 1691,
    optime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    optimeDate: ISODate('2025-04-15T12:10:47.000Z'),
    optimeWritten: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
    optimeWrittenDate: ISODate('2025-04-15T12:10:47.000Z'),
    lastAppliedWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    lastDurableWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    lastWrittenWallTime: ISODate('2025-04-15T12:10:47.851Z'),
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: 'Could not find member to sync from',
    electionTime: Timestamp({ t: 1744718937, i: 1 }),
    electionDate: ISODate('2025-04-15T12:08:57.000Z'),
    configVersion: 1,
    configTerm: 1,
    self: true,
    lastHeartbeatMessage: ''
  },
]
```

```
{
  _id: 1,
  name: 'localhost:27018',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 125,
  optime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeDurable: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeWritten: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeDate: ISODate('2025-04-15T12:10:47.000Z'),
  optimeDurableDate: ISODate('2025-04-15T12:10:47.000Z'),
  optimeWrittenDate: ISODate('2025-04-15T12:10:47.000Z'),
  lastAppliedWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastDurableWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastWrittenWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastHeartbeat: ISODate('2025-04-15T12:10:51.852Z'),
  lastHeartbeatRecv: ISODate('2025-04-15T12:10:52.825Z'),
  pingMs: Long('0'),
  lastHeartbeatMessage: '',
  syncSourceHost: 'localhost:27017',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 1,
  configTerm: 1
},
```

```
{
  _id: 2,
  name: 'localhost:27019',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 125,
  optime: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeDurable: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeWritten: { ts: Timestamp({ t: 1744719047, i: 1 }), t: Long('1') },
  optimeDate: ISODate('2025-04-15T12:10:47.000Z'),
  optimeDurableDate: ISODate('2025-04-15T12:10:47.000Z'),
  optimeWrittenDate: ISODate('2025-04-15T12:10:47.000Z'),
  lastAppliedWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastDurableWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastWrittenWallTime: ISODate('2025-04-15T12:10:47.851Z'),
  lastHeartbeat: ISODate('2025-04-15T12:10:51.852Z'),
  lastHeartbeatRecv: ISODate('2025-04-15T12:10:52.824Z'),
  pingMs: Long('0'),
  lastHeartbeatMessage: '',
  syncSourceHost: 'localhost:27017',
  syncSourceId: 0,
  infoMessage: '',
  configVersion: 1,
  configTerm: 1
}
```

## 4. Insertion Massive de Données et Traitement :

On utilise un script JavaScript dans mongosh pour insérer 10 000 faux utilisateurs (avec des données aléatoires).

```
use workshop ;

// Fonction pour générer des données aléatoires
function generateUser(id) {
  const roles = ["admin", "user", "editor", "guest"];

  return {
    _id: id,
    name: `User${id}`,
    email: `user${id}@example.com`,
```

```
age: Math.floor(Math.random() * 50 + 18),  
  
role: roles[Math.floor(Math.random() * roles.length)],  
  
createdAt: new Date()  
  
};  
  
}
```

```
rs0 [direct: primary] test> use workshop;  
switched to db workshop  
rs0 [direct: primary] workshop> function generateUser(id) {const roles = ["admin", "user", "editor", "guest"];return  
{ _id: id, name: `User${id}`, email: `user${id}@example.com`, age: Math.floor(Math.random() * 50 + 18), role: roles  
[Math.floor(Math.random() * roles.length)], createdAt: new Date() }; }  
[Function: generateUser]
```

// Insérer 10 000 utilisateurs en lots de 100 (optimisé pour MongoDB)

```
for (let i = 1; i <= 10000; i += 100) {  
  
  let bulk = [];  
  
  for (let j = i; j < i + 100; j++) {  
  
    bulk.push(generateUser(j));  
  
  }  
  
  db.users.insertMany(bulk);  
  
  print(`Inserted ${i + 99} users`);  
  
}
```

```
rs0 [direct: primary] workshop> for (let i = 1; i <= 10000; i += 100) { let bulk = []; for (let j = i; j < i + 100;  
j++) { bulk.push(generateUser(j)); } db.users.insertMany(bulk); print(`Inserted ${i + 99} users`);}  
Inserted 100 users  
Inserted 200 users  
Inserted 300 users  
Inserted 400 users  
Inserted 500 users  
Inserted 600 users  
Inserted 700 users  
Inserted 800 users  
Inserted 900 users  
Inserted 1000 users  
Inserted 1100 users  
Inserted 1200 users
```

On vérifie que les données sont insérées avec cette commande "db.users.countDocuments();" elle doit retourner 10000

## 5. Test de Réplication avec les données massives :

On va vérifier que les données sont copiées sur tous les nœuds.

On va se connecter à un nœud Secondary (port 27018)



```

C:\Users\user>mongosh --port 27018
Current Mongosh Log ID: 67ffb9a9ba2e63cdcdb71235
Connecting to:      mongodb://127.0.0.1:27018/?directConnect
ion=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.4.2
Using MongoDB:      8.0.5
Using Mongosh:      2.4.2
mongosh 2.5.0 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
  2025-04-16T15:05:41.440+01:00: Access control is not enabled
  for the database. Read and write access to data and configuratio
  n is unrestricted
  -----

rs0 [direct: secondary] test> |
  
```

Ensuite, on utilise la commande `rs.secondaryOk()`; pour autoriser la lecture sur la base de données.

On vérifie ensuite que les données de la base workshop sont bien accessibles en exécutant `use workshop`;

Puis `db.users.countDocuments()`; cette commande doit retourner la valeur 10000.

```

rs0 [direct: secondary] test> rs.secondaryOk();
DeprecationWarning: .setSecondaryOk() is deprecated. Use .setReadPref("primaryPreferred") instead
Setting read preference from "primary" to "primaryPreferred"

rs0 [direct: secondary] test> use workshop;
switched to db workshop
rs0 [direct: secondary] workshop> db.users.countDocuments();
10000
rs0 [direct: secondary] workshop> |
  
```

Pour trouver un user par son id, on utilise la commande `"db.users.findOne({ _id: 5000 });"`

```

rs0 [direct: secondary] workshop> db.users.findOne({ _id: 5000 }
);
{
  _id: 5000,
  name: 'User5000',
  email: 'user5000@example.com',
  age: 44,
  role: 'editor',
  createdAt: ISODate('2025-04-15T12:27:24.849Z')
}
  
```

## 6. Test de Réplication avec les données massives :

On va fermer la fenêtre terminale dans laquelle le mongod du Primary tourne (arrêter le serveur PRIMARY) avec **Ctrl + C**.

```

Invite de commandes
{"t":{"sdate":"2025-04-18T23:31:44.866+01:00"},"s":"I", "c":"WTRECOV", "id":22430, "ctx":"consoleTerminate", "msg":"WiredTiger message", "attr":{"message":{"ts_sec":1745015504, "ts_usec":866061, "thread":"15392:140730875979600", "session_name":"WT_CONNECTION.close", "category":"WT_VERB_RECOVERY_PROGRESS", "category_id":34, "verbose_level":"DEBUG_1", "verbose_level_id":1, "msg":"shutdown checkpoint has successfully finished and ran for 34 milliseconds"}}}}
{"t":{"sdate":"2025-04-18T23:31:44.870+01:00"},"s":"I", "c":"WTRECOV", "id":22430, "ctx":"consoleTerminate", "msg":"WiredTiger message", "attr":{"message":{"ts_sec":1745015504, "ts_usec":870057, "thread":"15392:140730875979600", "session_name":"WT_CONNECTION.close", "category":"WT_VERB_RECOVERY_PROGRESS", "category_id":34, "verbose_level":"DEBUG_1", "verbose_level_id":1, "msg":"shutdown was completed successfully and took 52ms, including 10ms for the rollback to stable, and 34ms for the checkpoint."}}}
{"t":{"sdate":"2025-04-18T23:31:45.026+01:00"},"s":"I", "c":"STORAGE", "id":4795901, "ctx":"consoleTerminate", "msg":"WiredTiger closed", "attr":{"durationMillis":216}}
{"t":{"sdate":"2025-04-18T23:31:45.102+01:00"},"s":"W", "c":"REPL", "id":6100702, "ctx":"ftdc", "msg":"Failed to get last stable recovery timestamp due to lock acquire timeout. Note this is expected if shutdown is in progress."}
{"t":{"sdate":"2025-04-18T23:31:45.104+01:00"},"s":"I", "c":"STORAGE", "id":22281, "ctx":"consoleTerminate", "msg":"shutdown: removing fs lock..."}
{"t":{"sdate":"2025-04-18T23:31:45.105+01:00"},"s":"I", "c":"-", "id":4784931, "ctx":"consoleTerminate", "msg":"Dropping the scope cache for shutdown"}
{"t":{"sdate":"2025-04-18T23:31:45.140+01:00"},"s":"I", "c":"FTDC", "id":20626, "ctx":"consoleTerminate", "msg":"Shutting down full-time diagnostic data capture"}
{"t":{"sdate":"2025-04-18T23:31:45.157+01:00"},"s":"I", "c":"CONTROL", "id":20565, "ctx":"consoleTerminate", "msg":"Now exiting"}
{"t":{"sdate":"2025-04-18T23:31:45.195+01:00"},"s":"I", "c":"CONTROL", "id":8423404, "ctx":"consoleTerminate", "msg":"mongod shutdown complete", "attr":{"summary of time elapsed":{"Enter terminal shutdown":"0 ms","Step down the replication coordinator for shutdown":"7 ms","Time spent in quiesce mode":"14993 ms","Shut down FLE Crud subsystem":"5 ms","Shut down MirrorMaestro":"5 ms","Shut down WaitForMajorityService":"10 ms","Shut down the logical session cache":"0 ms","Shut down the Query Analysis Sampler":"1 ms","Shut down the global connection pool":"3 ms","Shut down the flow control ticket holder":"0 ms","Shut down the replica set node executor":"1 ms","Shut down the thread that aborts expired transactions":"1 ms","Shut down the replica set aware services":"4 ms","Shut down replication":"0 ms","Shut down external state":"267 ms","Shut down replication executor":"0 ms","Join replication executor":"12 ms","Kill all operations for shutdown":"7 ms","Shut down all tenant migration access blockers on global shutdown":"23 ms","Shut down all open transactions":"2 ms","Acquire the RSTL for shutdown":"1 ms","Shut down the IndexBuildsCoordinator and wait for index builds to finish":"0 ms","Shut down the replica set monitor":"15 ms","Shut down the logical time validator":"1 ms","Shut down the migration utility executor":"8 ms","Shut down the transport layer":"2 ms","Shut down the health log":"0 ms","Shut down the TTL monitor":"1 ms","Shut down expired pre-images and documents removers":"1 ms","Shut down the storage engine":"305 ms","Wait for the oplog cap maintainer thread to stop":"0 ms","Shut down full-time data capture":"11 ms","Shut down online certificate status protocol manager":"0 ms","shutdownTask total elapsed time":"15778 ms"}}}}
{"t":{"sdate":"2025-04-18T23:31:45.240+01:00"},"s":"I", "c":"CONTROL", "id":23138, "ctx":"consoleTerminate", "msg":"Shutting down", "attr":{"exitCode":12}}
C:\Users\me\mongo-replica>
  
```

MongoDB va automatiquement élire un nouveau Primary parmi les deux Secondary. Cela peut prendre entre 5 à 15 secondes.

Pour vérifier le nouveau Primary, dans un autre terminal connectez-vous à un des autres serveurs mongosh --port 27018 :

```

mongosh mongodb://27.0.0.
Microsoft Windows [version 10.0.22631.4602]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\me>mongosh --port 27018
Current Mongosh Log ID: 6882d3a84ba07b47bc342c0e
Connecting to:   mongodb://127.0.0.1:27018/?directConnection=true&serverSelectionTimeoutMS=20000&appName=mongosh+2.1.5
Using MongoDB:  8.0.5
Using Mongosh:  2.1.5
mongosh 2.5.0 is available for download: https://www.mongodb.com/try/download/shell

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

-----
The server generated these startup warnings when booting
2025-04-18T21:54:43.649+01:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
  
```

Puis on vérifie le statut avec `rs.status()` :

```

mongosh mongodb://127.0.0.1:27018
{
  _id: 1,
  name: 'localhost:27018',
  health: 1,
  state: 1,
  stateStr: 'PRIMARY',
  uptime: 6054,
  optime: { ts: Timestamp({ t: 1745015726, i: 1 }), t: Long('2') },
  optimeDate: ISODate('2025-04-18T22:35:26.000Z'),
  optimeWritten: { ts: Timestamp({ t: 1745015726, i: 1 }), t: Long('2') },
  optimeWrittenDate: ISODate('2025-04-18T22:35:26.000Z'),
  lastAppliedWallTime: ISODate('2025-04-18T22:35:26.782Z'),
  lastDurableWallTime: ISODate('2025-04-18T22:35:26.782Z'),
  lastWrittenWallTime: ISODate('2025-04-18T22:35:26.782Z'),
  syncSourceHost: '',
  syncSourceId: -1,
  infoMessage: '',
  electionTime: Timestamp({ t: 1745014397, i: 1 }),
  electionDate: ISODate('2025-04-18T22:13:17.000Z'),
  configVersion: 1,
  configTerm: 2,
  self: true,
  lastHeartbeatMessage: ''
},
{
  _id: 2,
  name: 'localhost:27019',
  health: 1,
  state: 2,
  stateStr: 'SECONDARY',
  uptime: 3481,
  optime: { ts: Timestamp({ t: 1745015726, i: 1 }), t: Long('2') },
  optimeDurable: { ts: Timestamp({ t: 1745015726, i: 1 }), t: Long('2') },
  optimeWritten: { ts: Timestamp({ t: 1745015726, i: 1 }), t: Long('2') },
  optimeDate: ISODate('2025-04-18T22:35:26.000Z'),
}

```

On voit clairement dans la sortie :

```

{
  _id: 1,
  name: 'localhost:27018',
  stateStr: 'PRIMARY',
}

```

Cela signifie que le nœud sur le port 27018 est actuellement le PRIMARY.

On va insérer un autre objet :

```

use workshop;

db.users.insertOne({
  _id: 10001,
  name: "Karim",
  email: "Karim@example.com",
  age: 22,
  role: "editor",
  createdAt: new Date()
});

```

```
rs0 [direct: primary] workshop> db.users.insertOne({
...   _id: 10001,
...   name: "Karim",
...   email: "Karim@example.com",
...   age: 22,
...   role: "editor",
...   createdAt: new Date()
... });
{ acknowledged: true, insertedId: 10001 }
rs0 [direct: primary] workshop> |
```

Pour vérifier l'insertion `db.users.countDocuments();` // doit retourner 10001

```
rs0 [direct: primary] workshop> db.users.countDocuments();
10001
rs0 [direct: primary] workshop> |
```

On doit vérifier aussi sur un Secondary que les données ont été stockées.

On se connecte à le Secondary par `mongosh --port 27019` :

```
mongosh mongodb://127.0.0.1:27019
Microsoft Windows [version 10.0.22631.4602]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\me>mongosh --port 27019
Current Mongosh Log ID: 6802dbd1b8d3d960d5adfac6
Connecting to:      mongodb://127.0.0.1:27019/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.1.5
Using MongoDB:      8.0.5
Using Mongosh:       2.1.5
mongosh 2.5.0 is available for download: https://www.mongodb.com/try/download/shell

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

-----
The server generated these startup warnings when booting
2025-04-18T21:54:54.461+01:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

rs0 [direct: secondary] test> rs.secondaryOk();
DeprecationWarning: .setSecondaryOk() is deprecated. Use .setReadPref("primaryPreferred") instead
Setting read preference from "primary" to "primaryPreferred"

rs0 [direct: secondary] test> use workshop;
switched to db workshop
rs0 [direct: secondary] workshop> db.users.countDocuments();
10001
rs0 [direct: secondary] workshop> |
```

Il a retourné aussi 10001.