



INFORME DE TALLER DOCKER

I. PORTADA





Tema:	Práctica docker base de datos distribuidas
Unidad de Organización Curricular:	PROFESIONAL
Nivel y Paralelo:	Quinto “A”
Alumnos participantes:	Vallejo Rengifo John David
Asignatura:	Sistema de Base de Datos Distribuidos.
Docente:	Ing. Rubén Caiza, Mg.

II. DESARROLLO

PASO 1: ESTRUCTURA DE CARPETAS

Se crea una carpeta principal para organizar todos los archivos de la práctica.

Objetivo: Tener todos los archivos necesarios (configuración, datos, scripts) en un solo lugar.

Nombre	Fecha de modificación	Tipo	Tamaño
 departamentos	2/10/2025 15:24	Archivo de origen ...	1 KB
 docker-compose	2/10/2025 15:23	Archivo de origen ...	1 KB
 empleados	2/10/2025 15:25	Archivo de origen ...	1 KB
 ventas	2/10/2025 15:25	Archivo de origen ...	1 KB

PASO 2: LEVANTAR LOS CONTENORES

Se ejecuta el siguiente comando en la terminal, desde la carpeta donde está el docker-compose.yml:

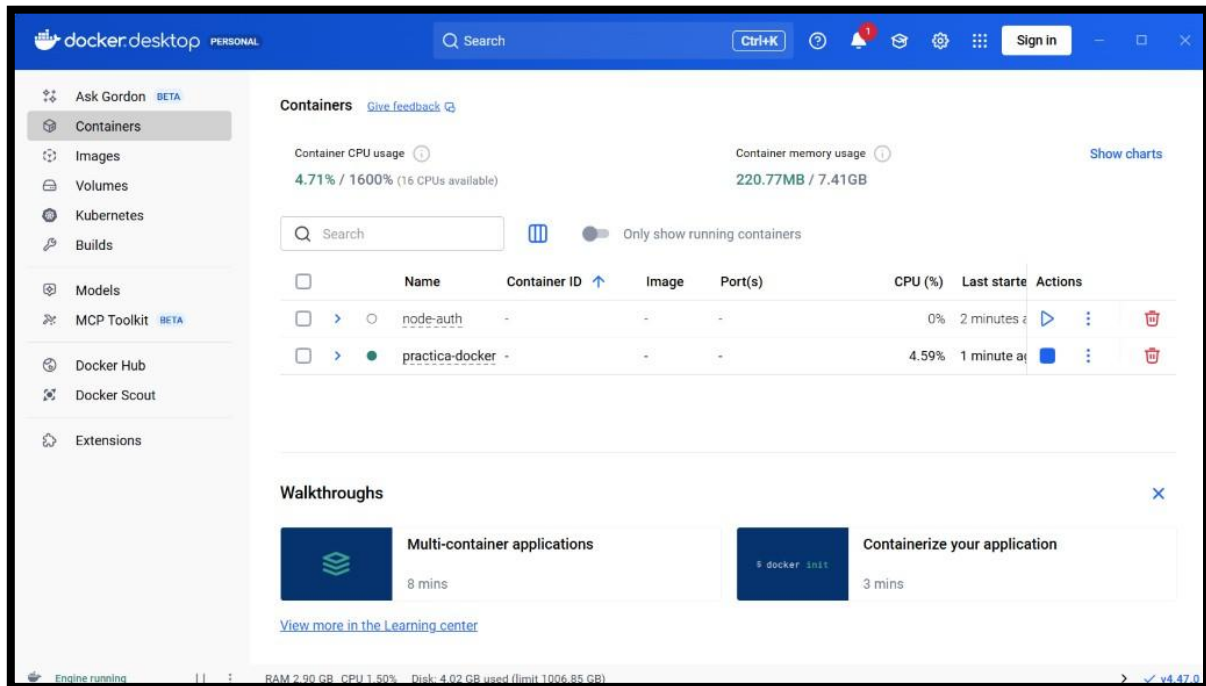
Comando: docker compose up -d

```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker compose up -d
time="2025-10-02T17:56:05-05:00" level=warning msg="C:\\Users\\Lenovo\\Desktop\\QUINTO\\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\\Practica-Docker\\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 11/11
  ✓ mongo3 Pulled                                42.2s
  ✓ mongo1 Pulled                                42.3s
  ✓ mongo2 Pulled                                42.2s
  ✓ 108411e1d459 Pull complete                    8.0s
  ✓ c4d0a7bd57e7 Pull complete                    15.1s
  ✓ bf76f00fdecd Pull complete                    8.8s
  ✓ 57d7c37a3a3b Pull complete                    37.9s
  ✓ a46b1823f120 Pull complete                    18.5s
  ✓ dfad3838f14f Pull complete                    17.3s
  ✓ af6eca94c810 Pull complete                    12.7s
  ✓ 04ceb0169a10 Pull complete                    8.2s
[+] Running 7/7
  ✓ Network practica-docker_default Created          0.0s
  ✓ Volume practica-docker_mongo1 Created           0.0s
  ✓ Volume practica-docker_mongo2 Created           0.0s
  ✓ Volume practica-docker_mongo3 Created           0.0s
  ✓ Container mongo3 Started                       0.4s
  ✓ Container mongo2 Started                       0.4s
  ✓ Container mongo1 Started                       0.4s
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> |
```



PASO 3: VERIFICAR LOS CONTENEDORES

En la interfaz de docker desktop se verifica que se hayan creado correctamente los contenedores y que estén en ejecución.



PASO 4: INICIAR EL REPLICA SET

Objetivo: Establecer mongo1 como primario y mongo2 y mongo3 como secundarios en el replica set rs0.

Resultado: El replica set comienza a operar con un primario y dos secundarios.

```
docker exec -it mongo1 mongosh --eval '
rs.initiate({
  _id: "rs0",
  members: [
    { _id: 0, host: "mongo1:27017"},
    { _id: 1, host: "mongo2:27017"},
    { _id: 2, host: "mongo3:27017"}
  ]
})'
```



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker exec -it mongo1 mongosh
Current Mongosh Log ID: 68df047fa9416ca6e5ce5f46
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2
.5.8
Using MongoDB:      7.0.25
Using Mongosh:       2.5.8

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

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2025-10-02T22:56:48.685+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-10-02T22:56:48.803+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2025-10-02T22:56:48.803+00:00: vm.max_map_count is too low
-----

test> rs.initiate({
...   _id: "rs0"
...   members: [
...     { _id: 0, host: "mongo1:27017" },
...     { _id: 1, host: "mongo2:27017" },
...     { _id: 2, host: "mongo3:27017" }
...   ]
... })
{ ok: 1 }
rs0 [direct: other] test> |
```

PASO 5: VERIFICAR EL ESTADO DEL REPLICA SET

Objetivo: Verificar que mongo1 es PRIMARY, y mongo2 y mongo3 son SECONDARY.

Resultado: Salida en formato JSON que muestra el estado de cada miembro del replica set. docker

exec -it mongo1 mongosh --eval 'rs.status()'

```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker exec -it mongo1 mongosh --eval 'rs.status()'
{
  set: 'rs0',
  date: ISODate('2025-10-02T23:04:43.117Z'),
  myState: 1,
  term: Long('1'),
  syncSourceHost: '',
  syncSourceId: -1,
  heartbeatIntervalMillis: Long('2000'),
  majorityVoteCount: 2,
  writeMajorityCount: 2,
  votingMembersCount: 3,
  writableVotingMembersCount: 3,
  optimes: {
    lastCommittedOpTime: { ts: Timestamp({ t: 1759446279, i: 1 }), t: Long('1') },
    lastCommittedWallTime: ISODate('2025-10-02T23:04:39.058Z'),
    readConcernMajorityOpTime: { ts: Timestamp({ t: 1759446279, i: 1 }), t: Long('1') },
    appliedOpTime: { ts: Timestamp({ t: 1759446279, i: 1 }), t: Long('1') },
    durableOpTime: { ts: Timestamp({ t: 1759446279, i: 1 }), t: Long('1') },
    lastAppliedWallTime: ISODate('2025-10-02T23:04:39.058Z'),
    lastDurableWallTime: ISODate('2025-10-02T23:04:39.058Z')
  },
  lastStableRecoveryTimestamp: Timestamp({ t: 1759446269, i: 1 }),
  electionCandidateMetrics: {
    lastElectionReason: 'electionTimeout',
    lastElectionDate: ISODate('2025-10-02T23:02:49.014Z'),
    electionTerm: Long('1'),
    lastCommittedOpTimeAtElection: { ts: Timestamp({ t: 1759446158, i: 1 }), t: Long('-1') },
    lastSeenOpTimeAtElection: { ts: Timestamp({ t: 1759446158, i: 1 }), t: Long('-1') },
    numVotesNeeded: 2,
    priorityAtElection: 1,
    electionTimeoutMillis: Long('10000'),
    numCatchUpOps: Long('0'),
    newTermStartDate: ISODate('2025-10-02T23:02:49.050Z'),
    wMajorityWriteAvailabilityDate: ISODate('2025-10-02T23:02:49.574Z')
  },
}
```



PASO 6: IMPORTAR LOS DATOS

Objetivo: Cargar los datos de ejemplo en las colecciones correspondientes de la base de datos escuela.

Resultado: Las colecciones departamentos, empleados y ventas existen en la base de datos escuela y contienen los datos importados.

```
docker exec -i mongo1 mongoimport --db escuela --collection departamentos --file
/dev/stdin < departamentos.json
docker exec -i mongo1 mongoimport --db escuela --collection empleados --file
/dev/stdin < empleados.json
docker exec -i mongo1 mongoimport --db escuela --collection ventas --file
/dev/stdin < ventas.json
```

```
Lenovo@Teo MINGW64 ~/Desktop/QUINTO/SISTEMAS DE BASES DE DATOS DISTRIBUIDOS/Prac
tica-Docker
$ docker exec -i mongo1 mongoimport --db escuela --collection departamentos --fi
le /dev/stdin < departamentos.json
2025-10-02T23:11:10.428+0000    connected to: mongodb://localhost/
2025-10-02T23:11:10.447+0000    4 document(s) imported successfully. 0 document(
s) failed to import.
```

```
Lenovo@Teo MINGW64 ~/Desktop/QUINTO/SISTEMAS DE BASES DE DATOS DISTRIBUIDOS/Prac
tica-Docker
$ docker exec -i mongo1 mongoimport --db escuela --collection empleados --file /dev/stdin < empleados.json
2025-10-02T23:12:35.528+0000    connected to: mongodb://localhost/
2025-10-02T23:12:35.548+0000    6 document(s) imported successfully. 0 document(s) failed to import.
```

```
Lenovo@Teo MINGW64 ~/Desktop/QUINTO/SISTEMAS DE BASES DE DATOS DISTRIBUIDOS/Practica-Docker
$ docker exec -i mongo1 mongoimport --db escuela --collection ventas --file /dev/stdin < ventas.json
2025-10-02T23:12:59.628+0000    connected to: mongodb://localhost/
2025-10-02T23:12:59.648+0000    6 document(s) imported successfully. 0 document(s) failed to import.
```

PASO 7: Verificar que los datos se hayan importado correctamente

Objetivo: Confirmar que las colecciones empleados, departamentos y ventas en la base de datos escuela contienen la cantidad correcta de documentos después de la importación.

```
docker exec -it mongo1 mongosh
> use escuela
> db.empleados.countDocuments()
6
> db.departamentos.countDocuments()
4
> db.ventas.countDocuments()
6
```



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker exec -it mongol mongosh
Current Mongosh Log ID: 68df073c2a34b511f5ce5f46
Connecting to: mongod://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB: 7.0.25
Using Mongosh: 2.5.8

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

-----
The server generated these startup warnings when booting
2025-10-02T22:56:48.685+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/pri
odnotes-filesystem
2025-10-02T22:56:48.803+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2025-10-02T22:56:48.803+00:00: vm.max_map_count is too low
-----

rs0 [direct: primary] test> use escuela
switched to db escuela
rs0 [direct: primary] escuela> db.empleados.countDocuments()
6
rs0 [direct: primary] escuela> db.departamentos.countDocuments()
4
rs0 [direct: primary] escuela> db.ventas.countDocuments()
6
```

PASO 8: EJECUTAR LAS CONSULTAS

Objetivo: Demostrar el uso de Aggregation Pipeline con operadores como \$lookup, \$group, \$match, \$project, \$sort, \$limit, \$setWindowFields, etc.

Resultado: Cada consulta devuelve los resultados esperados. Se toman capturas de pantalla de los resultados para la entrega.

C1: Empleados con salario mayor al promedio de la empresa

Método 1 (con ventanas, MongoDB ≥ 5):

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       // No es necesario especificar 'window' aquí para operar sobre todos los docs
...       output: {
...         promEmpresa: {
...           $avg: "$salario"
...         }
...       }
...     },
...     $match: {
...       salario: { $gt: "$promEmpresa" }
...     },
...     $project: {
...       nombre: 1,
...       salario: 1,
...       promEmpresa: 1
...     }
...   }
... ])
```




Método 2 (sin ventanas):

```
rs0 [direct: primary] escuela> var promedio = db.empleados.aggregate([
...   { $group: { _id: null, prom: { $avg: "$salario" } } }
... ]).toArray()[0].prom;
...
... db.empleados.aggregate([
...   {
...     $match: {
...       salario: { $gt: promedio }
...     }
...   },
...   {
...     $project: {
...       nombre: 1,
...       salario: 1
...     }
...   }
... ])
[
  { _id: 1, nombre: 'Ana', salario: 1200 },
  { _id: 6, nombre: 'Frank', salario: 2000 },
  { _id: 3, nombre: 'Carla', salario: 1500 }
]
```

C2: Departamentos sin empleados asignados

```
rs0 [direct: primary] escuela> db.departamentos.aggregate([
...   {
...     $lookup: {
...       from: "empleados",
...       localField: "_id",
...       foreignField: "departamento_id",
...       as: "empleados"
...     }
...   },
...   {
...     $addFields: {
...       countEmpleados: { $size: "$empleados" }
...     }
...   },
...   {
...     $match: {
...       countEmpleados: 0
...     }
...   },
...   {
...     $project: {
...       nombre: 1,
...       countEmpleados: 1
...     }
...   }
... ])
[ { _id: 4, nombre: 'Logística', countEmpleados: 0 } ]
```



C3: Empleado con salario más alto
Opción A (sort + limit):

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   { $sort: { salario: -1 } },
...   { $limit: 1 },
...   {
...     $project: {
...       nombre: 1,
...       salario: 1
...     }
...   }
... ])
...
[ { _id: 6, nombre: 'Frank', salario: 2000 } ]
```

Opción B (group + max):

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $group: {
...       _id: null,
...       maxSalario: { $max: "$salario" }
...     }
...   },
...   {
...     $lookup: {
...       from: "empleados",
...       localField: "maxSalario",
...       foreignField: "salario",
...       as: "empleadoMax"
...     }
...   },
...   {
...     $unwind: "$empleadoMax"
...   },
...   {
...     $replaceRoot: { newRoot: "$empleadoMax" }
...   },
...   {
...     $project: {
...       nombre: 1,
...       salario: 1
...     }
...   }
... ])
...
[ { _id: 6, nombre: 'Frank', salario: 2000 } ]
```



C4: Para cada empleado, mostrar el salario promedio de su departamento Opción con ventanas:

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       partitionBy: "$departamento_id",
...       output: {
...         promDepartamento: { $avg: "$salario" }
...       }
...     }
...   },
...   {
...     $project: {
...       nombre: 1,
...       salario: 1,
...       departamento_id: 1,
...       promDepartamento: 1
...     }
...   }
... ])
[
  {
    _id: 1,
    nombre: 'Ana',
    salario: 1200,
    departamento_id: 1,
    promDepartamento: 1050
  },
  {
    _id: 2,
    nombre: 'Bruno',
    salario: 900,
    departamento_id: 1,
    promDepartamento: 1050
  },
  {
    _id: 4,
    nombre: 'Diego',
    salario: 800,
    departamento_id: 2,
    promDepartamento: 1433.3333333333333
  },
  {
    _id: 6,
    nombre: 'Frank',
    salario: 2000,
    departamento_id: 2,
```




Opción sin ventanas:

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $lookup: {
...       from: "empleados",
...       let: { depId: "$departamento_id" },
...       pipeline: [
...         {
...           $match: {
...             $expr: { $eq: ["$departamento_id", "$$depId"] }
...           }
...         },
...         {
...           $group: {
...             _id: null,
...             prom: { $avg: "$salario" }
...           }
...         }
...       ],
...       as: "promDep"
...     }
...   },
...   {
...     $addFields: {
...       promDepartamento: { $arrayElemAt: ["$promDep.prom", 0] }
...     }
...   },
...   {
...     $project: {
...       nombre: 1,
...       salario: 1,
...       departamento_id: 1,
...       promDepartamento: 1
...     }
...   }
... ])
[
  {
    _id: 1,
    nombre: 'Ana',
    salario: 1200,
    departamento_id: 1,
    promDepartamento: 1050
  },
  {
    _id: 4,
    nombre: 'Diego',
    salario: 800,
```



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



C5: Departamentos cuyo promedio salarial es mayor al promedio general
Solución 1 (ventanas):

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       // No es necesario especificar 'window' aquí para operar sobre todos los docs
...       output: {
...         promGeneral: { $avg: "$salario" }
...       }
...     },
...     {
...       $group: {
...         _id: "$departamento_id",
...         promDepartamento: { $avg: "$salario" },
...         promGeneral: { $first: "$promGeneral" }
...       }
...     },
...     {
...       $match: {
...         $expr: { $gt: ["$promDepartamento", "$promGeneral"] }
...       }
...     },
...     {
...       $lookup: {
...         from: "departamentos",
...         localField: "_id",
...         foreignField: "_id",
...         as: "depto"
...       }
...     },
...     {
...       $unwind: "$depto"
...     },
...     {
...       $project: {
...         nombreDepartamento: "$depto.nombre",
...         promDepartamento: 1,
...         promGeneral: 1
...       }
...     }
...   ]
... ]
[
  {
    _id: 2,
    promDepartamento: 1433.3333333333333,
    promGeneral: 1183.3333333333333,
    nombreDepartamento: 'TI'
  }
]
```



Solución 2 (doble pipeline):

```
rs0 [direct: primary] escuela> var promGeneral = db.empleados.aggregate([
...   { $group: { _id: null, prom: { $avg: "$salario" } } }
... ]).toArray()[0].prom;
...
... db.empleados.aggregate([
...   {
...     $group: {
...       _id: "$departamento_id",
...       promDepartamento: { $avg: "$salario" }
...     }
...   },
...   {
...     $match: {
...       promDepartamento: { $gt: promGeneral }
...     }
...   },
...   {
...     $lookup: {
...       from: "departamentos",
...       localField: "_id",
...       foreignField: "_id",
...       as: "depto"
...     }
...   },
...   {
...     $unwind: "$depto"
...   },
...   {
...     $project: {
...       nombreDepartamento: "$depto.nombre",
...       promDepartamento: 1
...     }
...   }
... ])
[
  {
    _id: 2,
    promDepartamento: 1433.3333333333333,
    nombreDepartamento: 'TI'
  }
]
```



C6: Ventas: sucursal “top” por mes

```
rs0 [direct: primary] escuela> db.ventas.aggregate([
... {
...   $group: {
...     _id: {
...       mes: "$_id.mes", // Extraemos el mes del objeto anidado
...       sucursal: "$_id.sucursal" // Extraemos la sucursal del objeto anidado
...     },
...     total: { $sum: "$total" } // Sumamos el total de ventas
...   },
... },
... {
...   $sort: { "_id.mes": 1, total: -1 } // Ordenamos por mes ascendente y total descendente
... },
... {
...   $group: {
...     _id: "$_id.mes", // Agrupamos de nuevo por mes
...     topSucursal: { $first: "$_id.sucursal" }, // Tomamos la primera sucursal (la de mayor total)
...     total: { $first: "$total" } // Tomamos su total
...   },
... },
... {
...   $project: {
...     mes: "$_id", // Mostramos el mes
...     sucursal: "$topSucursal", // Mostramos la sucursal ganadora
...     total: 1 // Mostramos el total
...   },
... },
... {
...   $sort: { mes: 1 } // Ordenamos por mes
... },
... ])
[
  {
    _id: '2025-08',
    total: 42000,
    mes: '2025-08',
    sucursal: 'Guayaquil'
  },
  { _id: '2025-09', total: 39000, mes: '2025-09', sucursal: 'Quito' }
]
```

PASO 9: PRUEBA DE RESILIENCIA

Objetivo: Simular la caída de un nodo secundario para probar la resiliencia del replica set.

Resultado: La consulta se ejecuta correctamente, ya que el primario (mongo1) sigue operativo. El sistema sigue funcionando normalmente para lectura y escritura.

```
docker stop mongo3
docker exec -it mongo1 mongosh
> use escuela
> db.empleados.countDocuments()
>
```



```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker stop mongo3
mongo3
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker exec -it mongoi mongosh
Current Mongosh Log ID: 68df0ddBeaf687ale9ce5f46
Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB: 7.0.25
Using Mongosh: 2.5.8

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

-----
The server generated these startup warnings when booting
2025-10-02T22:56:48.685+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-10-02T22:56:48.803+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2025-10-02T22:56:48.803+00:00: vm.max_map_count is too low
-----

rs0 [direct: primary] test> use escuela
switched to db escuela
rs0 [direct: primary] escuela> db.empleados.aggregate([ ... ])
Uncaught:
SyntaxError: Unexpected token {1:29}

> | db.empleados.aggregate([ ... ])
  | ^
  |
  |
  |
rs0 [direct: primary] escuela> db.empleados.countDocuments()
6
```

Explicación: MongoDB no necesita 3 nodos para operar. Con 2 nodos activos (1 PRIMARY + 1 SECONDARY), puede seguir funcionando normalmente, siempre y cuando el primario no se caiga. Si se cae el primario, se realizaría una elección entre los nodos restantes (si hay mayoría).

```
docker start mongo3
```

```
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker> docker start mongo3
mongo3
PS C:\Users\Lenovo\Desktop\QUINTO\SISTEMAS DE BASES DE DATOS DISTRIBUIDOS\Practica-Docker>
```