INFORME DE TALLER DOCKER

I. PORTADA

Tema:	Implementación y configuración de un clúster de MongoDB con Docker y Replica Set
Unidad de Organización Curricular:	PROFESIONAL
Nivel y Paralelo:	Quinto "A"
Alumnos participantes:	Aldas Jordan Wellington Ismael Caguasango Bayas Alex Patricio Gómez Llerena Luis Fernando Paredes Barrera Luis Enrique
Asignatura:	Sistemas de Base de Datos Distribuidas
Docente:	Ing. Jose Ruben Caiza Caizabuano, Mg.

II. INFORME DE TALLER DOCKER

2.1 Objetivos

2.1.1 Objetivo General:

Implementar un clúster de MongoDB utilizando Docker para comprender el funcionamiento de la replicación, la resiliencia y la gestión de datos en entornos distribuidos.

2.1.2 Objetivos Específicos:

- Configurar los contenedores de Docker necesarios para levantar el clúster de MongoDB con Replica Set.
- Importar y gestionar los datos dentro del entorno configurado, verificando su correcta sincronización entre los nodos.
- Evaluar la resiliencia del clúster mediante pruebas de falla y recuperación de nodos.

2.2 Modalidad

Presencial

2.3 Tiempo de duración

Presenciales: 3

No presenciales: 0

2.4 Instrucciones

Para la realización de este taller, primero se debe preparar el entorno de trabajo instalando Docker y verificando que funcione correctamente. Luego, es necesario crear una red personalizada que permita la comunicación entre los contenedores que formarán parte del clúster. Posteriormente, se procede a configurar los tres contenedores de MongoDB (mongo1, mongo2 y mongo3), los cuales conformarán el Replica Set. Desde el contenedor principal

(mongo1), se debe inicializar la replicación y verificar su correcto estado mediante el comando rs.status().

Una vez establecida la configuración, se copian los archivos .json al interior del contenedor utilizando la ruta /tmp, para luego ejecutar el comando mongoimport e importar los datos dentro de la base de datos. Después de la importación, se accede a la shell de MongoDB e ingresa a la base de datos con el comando use escuela, verificando que los datos estén almacenados correctamente.

Con el entorno en funcionamiento, se pueden ejecutar las consultas requeridas, como por ejemplo: el empleado con el salario más alto, los departamentos sin empleados asignados, la sucursal con mayores ventas o el cálculo del salario promedio por departamento. Estas consultas se pueden realizar tanto con funciones de ventana como sin ellas, dependiendo de la versión de MongoDB y las funcionalidades disponibles.

Finalmente, se lleva a cabo una prueba de resiliencia deteniendo uno de los nodos del clúster para observar el comportamiento del sistema. Con el comando rs.status() se verifica que el nodo detenido no esté disponible y se confirma que el clúster sigue operativo mientras el nodo primario permanezca activo.

En caso de que el nodo principal se apague, se debe comprobar que otro nodo tome el rol de primario de forma automática, garantizando la continuidad del servicio.

2.5 Listado de equipos, materiales y recursos

Listado de equipos y materiales generales empleados en la guía práctica:

- Computador con Windows/Linux/macOS.
- MongoDB
- Docker

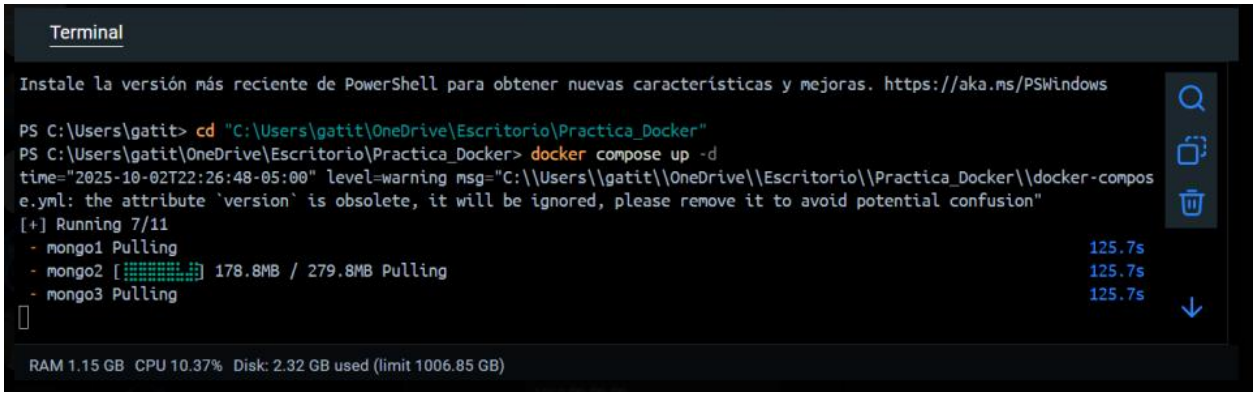
TAC (Tecnologías para el Aprendizaje y Conocimiento) empleados en la guía práctica:

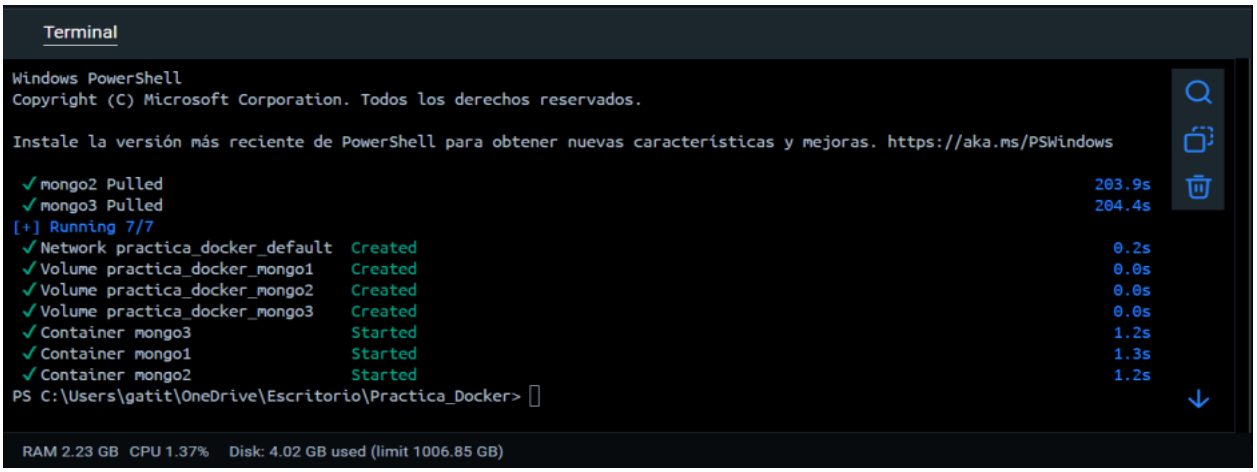
☒Plataformas educativas
☒Simuladores y laboratorios virtuales
☐Aplicaciones educativas
☒Recursos audiovisuales
☐Gamificación
☐Inteligencia Artificial
Otros (Especifique): _____

2.6 Actividades desarrolladas

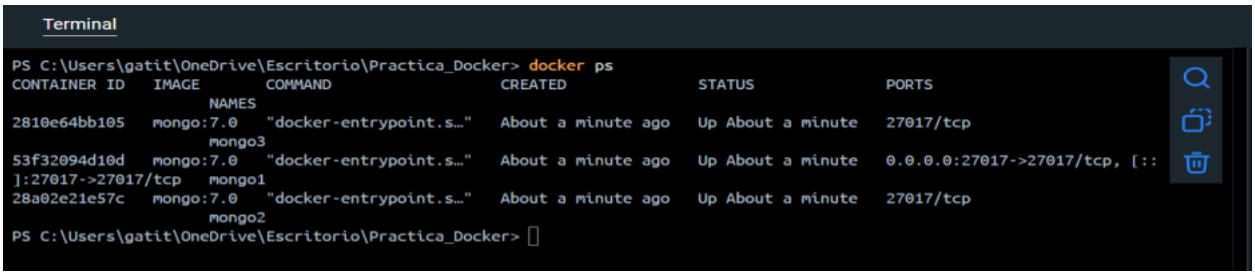
TALLER DOCKER

EJECUTAR Y LEVANTAR EL CLÚSTER DE MONGODB



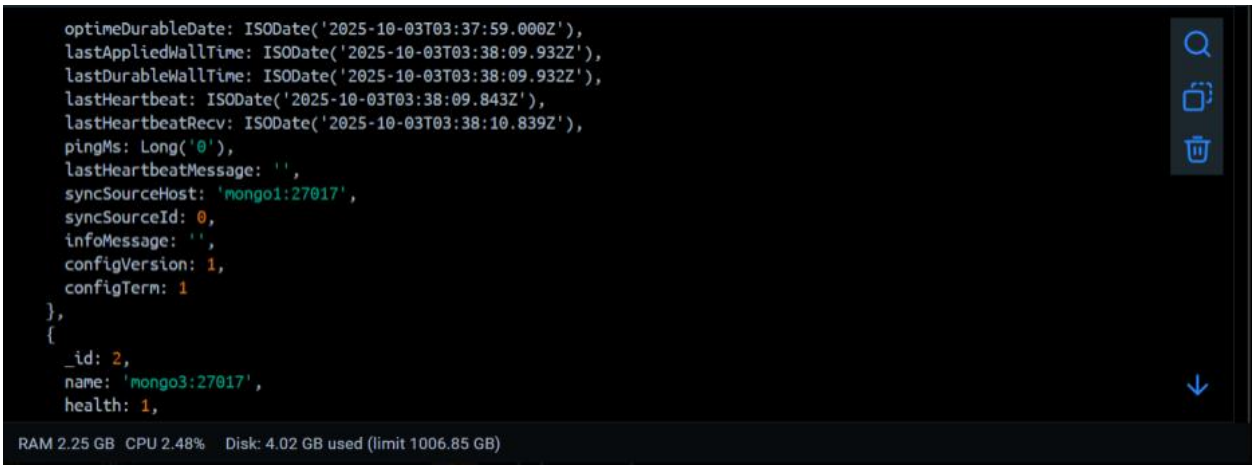


VERIFICAR QUE LOS CONTENEDORES ESTÉN ACTIVOS



VERIFICAR QUE SE HAYA CONFIGURADO CORRECTAMENTE


electionTime: Timestamp({ t: 1759462649, i: 1 }),
electionDate: ISODate('2025-10-03T03:37:29.000Z'),
configVersion: 1,
configTerm: 1,
self: true,
lastHeartbeatMessage: ''
},
{
_id: 1,
name: 'mongo2:27017',
health: 1,
state: 2,
stateStr: 'SECONDARY',
uptime: 52,
optime: { ts: Timestamp({ t: 1759462679, i: 1 }), t: Long('1') },
optimeDurable: { ts: Timestamp({ t: 1759462679, i: 1 }), t: Long('1') },
optimeDate: ISODate('2025-10-03T03:37:59.000Z'),
RAM 2.24 GB CPU 0.50% Disk: 4.02 GB used (limit 1006.85 GB)


optimeDurableDate: ISODate('2025-10-03T03:37:59.000Z'),
lastAppliedWallTime: ISODate('2025-10-03T03:38:09.932Z'),
lastDurableWallTime: ISODate('2025-10-03T03:38:09.932Z'),
lastHeartbeat: ISODate('2025-10-03T03:38:09.843Z'),
lastHeartbeatRecv: ISODate('2025-10-03T03:38:10.839Z'),
pingMs: Long('0'),
lastHeartbeatMessage: '',
syncSourceHost: 'mongo1:27017',
syncSourceId: 0,
infoMessage: '',
configVersion: 1,
configTerm: 1
},
{
_id: 2,
name: 'mongo3:27017',
health: 1,
RAM 2.25 GB CPU 2.48% Disk: 4.02 GB used (limit 1006.85 GB)

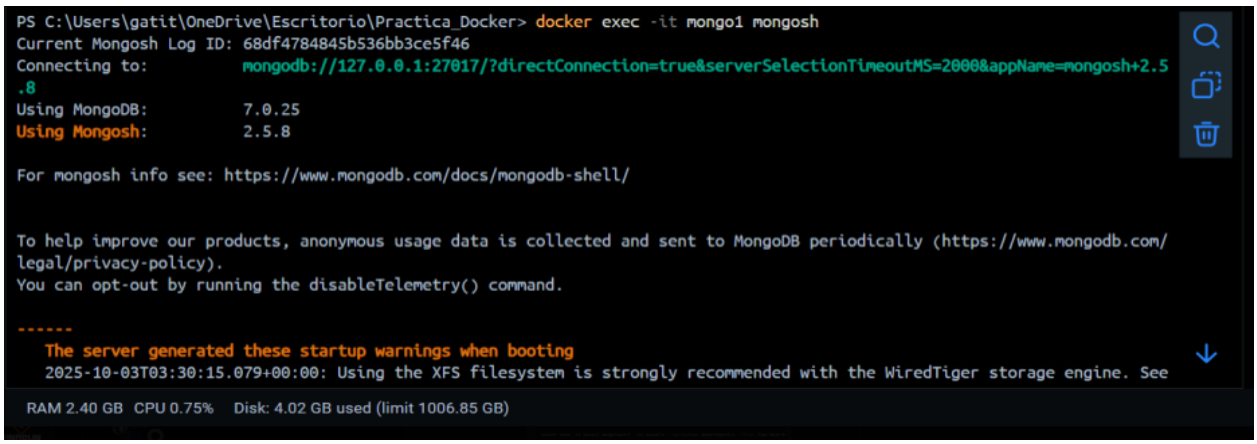

state: 2,
stateStr: 'SECONDARY',
uptime: 52,
optime: { ts: Timestamp({ t: 1759462679, i: 1 }), t: Long('1') },
optimeDurable: { ts: Timestamp({ t: 1759462679, i: 1 }), t: Long('1') },
optimeDate: ISODate('2025-10-03T03:37:59.000Z'),
optimeDurableDate: ISODate('2025-10-03T03:37:59.000Z'),
lastAppliedWallTime: ISODate('2025-10-03T03:38:09.932Z'),
lastDurableWallTime: ISODate('2025-10-03T03:38:09.932Z'),
lastHeartbeat: ISODate('2025-10-03T03:38:09.834Z'),
lastHeartbeatRecv: ISODate('2025-10-03T03:38:10.839Z'),
pingMs: Long('0'),
lastHeartbeatMessage: '',
syncSourceHost: 'mongo1:27017',
syncSourceId: 0,
infoMessage: '',
configVersion: 1,
M 2.25 GB CPU 1.00% Disk: 4.02 GB used (limit 1006.85 GB)

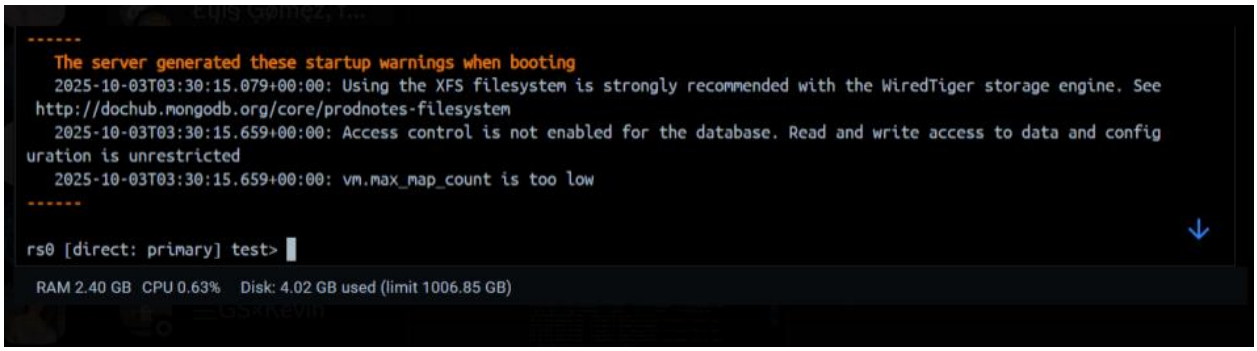

configTerm: 1
}
],
ok: 1,
'$clusterTime': {
clusterTime: Timestamp({ t: 1759462689, i: 1 }),
signature: {
hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
keyId: Long('0')
}
},
operationTime: Timestamp({ t: 1759462689, i: 1 })
}
PS C:\Users\gatit\OneDrive\Escritorio\Practica_Docker>
RAM 2.25 GB CPU 2.50% Disk: 4.02 GB used (limit 1006.85 GB)

COPIAMOS LOS ARCHIVOS. JSON AL CONTENEDOR ANTES DE IMPORTARLOS

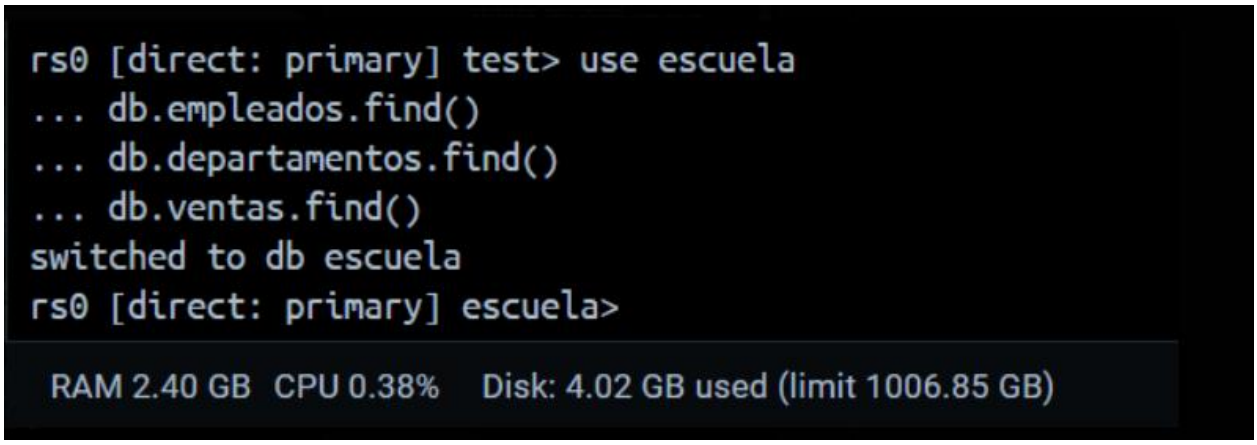
EJECUTAMOS MONGONIMPORT DENTRO DEL CONTENEDOR A UTILIZAR CON LA RUTA /TMP

INGRESAMOS A LA SHELL DE MONGODB

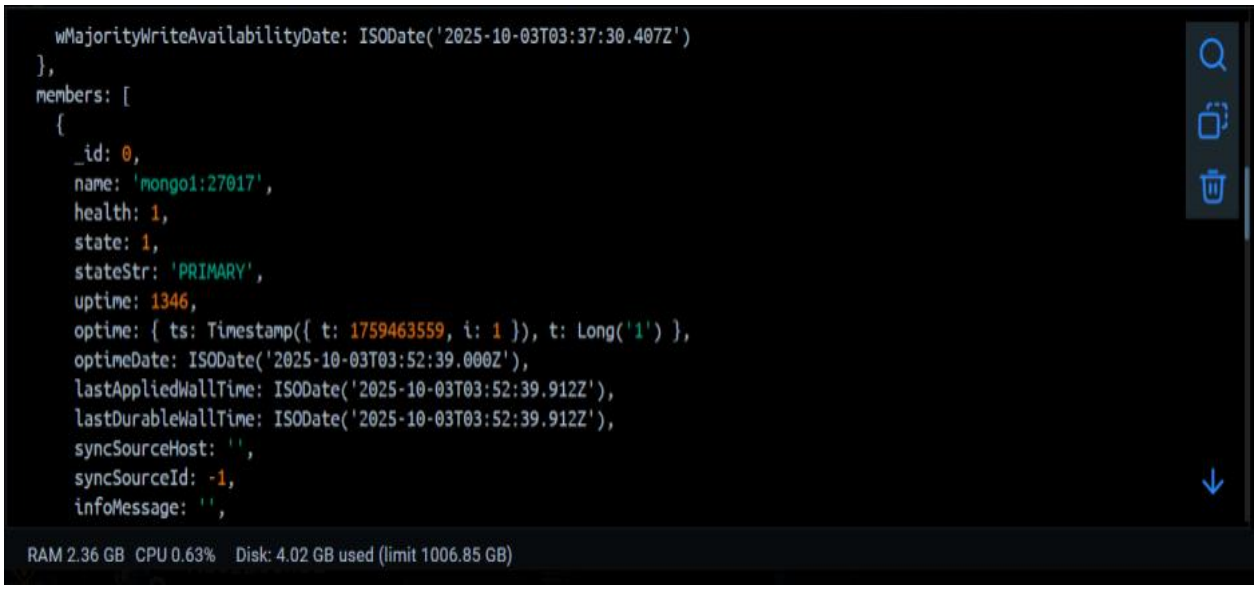


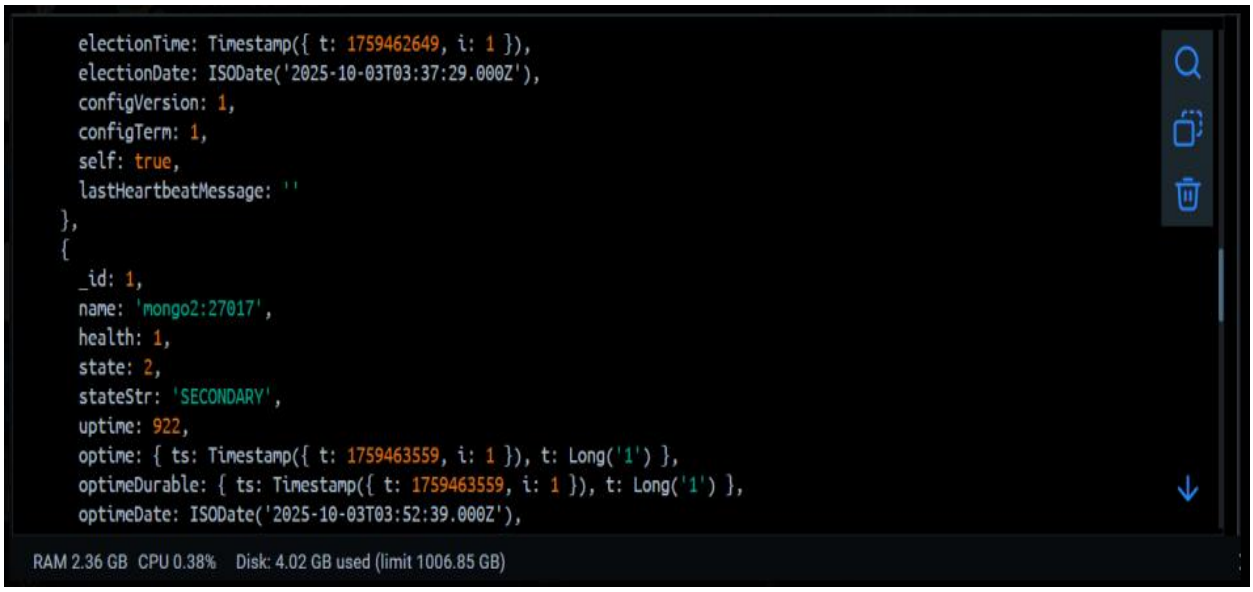


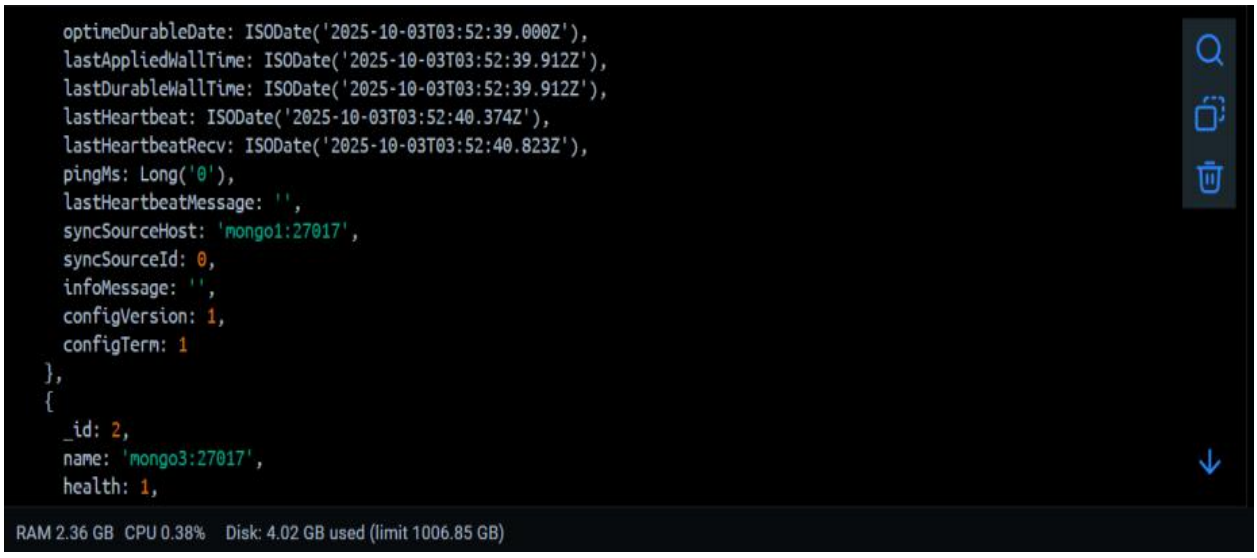
Una vez dentro, con el comando **use escuela (que será el nombre de la base de datos),** se podrá visualizar los datos que están guardados.

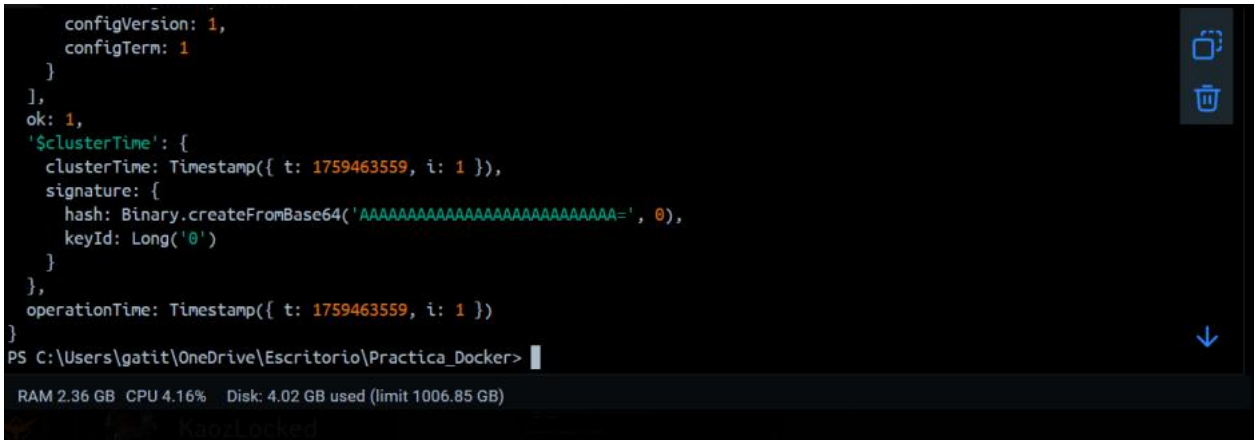


Luego revisamos el estado de REPLICA SET inicializado.


wMajorityWriteAvailabilityDate: ISODate('2025-10-03T03:37:30.407Z')
},
members: [
{
_id: 0,
name: 'mongo1:27017',
health: 1,
state: 1,
stateStr: 'PRIMARY',
uptime: 1346,
optime: { ts: Timestamp({ t: 1759463559, i: 1 }), t: Long('1') },
optimeDate: ISODate('2025-10-03T03:52:39.000Z'),
lastAppliedWallTime: ISODate('2025-10-03T03:52:39.912Z'),
lastDurableWallTime: ISODate('2025-10-03T03:52:39.912Z'),
syncSourceHost: '',
syncSourceId: -1,
infoMessage: '',
RAM 2.36 GB CPU 0.63% Disk: 4.02 GB used (limit 1006.85 GB)


electionTime: Timestamp({ t: 1759462649, i: 1 }),
electionDate: ISODate('2025-10-03T03:37:29.000Z'),
configVersion: 1,
configTerm: 1,
self: true,
lastHeartbeatMessage: ''
},
{
_id: 1,
name: 'mongo2:27017',
health: 1,
state: 2,
stateStr: 'SECONDARY',
uptime: 922,
optime: { ts: Timestamp({ t: 1759463559, i: 1 }), t: Long('1') },
optimeDurable: { ts: Timestamp({ t: 1759463559, i: 1 }), t: Long('1') },
optimeDate: ISODate('2025-10-03T03:52:39.000Z'),
RAM 2.36 GB CPU 0.38% Disk: 4.02 GB used (limit 1006.85 GB)


optimeDurableDate: ISODate('2025-10-03T03:52:39.000Z'),
lastAppliedWallTime: ISODate('2025-10-03T03:52:39.912Z'),
lastDurableWallTime: ISODate('2025-10-03T03:52:39.912Z'),
lastHeartbeat: ISODate('2025-10-03T03:52:40.374Z'),
lastHeartbeatRecv: ISODate('2025-10-03T03:52:40.823Z'),
pingMs: Long('0'),
lastHeartbeatMessage: '',
syncSourceHost: 'mongo1:27017',
syncSourceId: 0,
infoMessage: '',
configVersion: 1,
configTerm: 1
},
{
_id: 2,
name: 'mongo3:27017',
health: 1,
RAM 2.36 GB CPU 0.38% Disk: 4.02 GB used (limit 1006.85 GB)



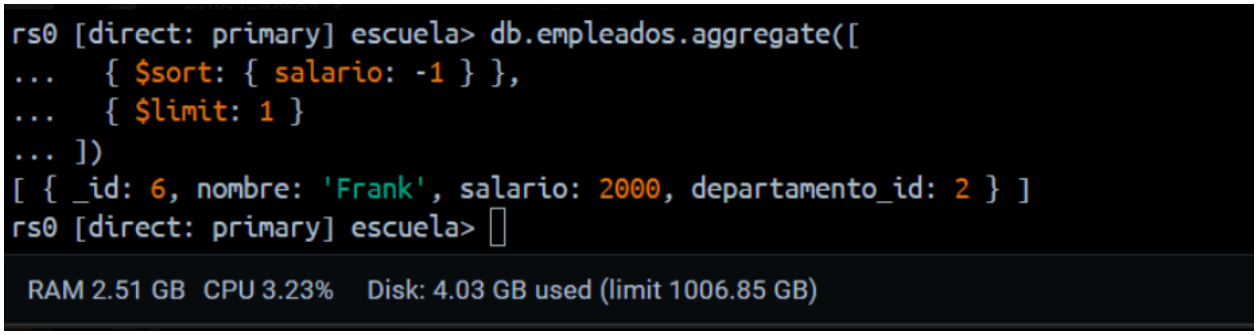
EJECUCION DE CONSULTAS.

Para la ejecución de consultas, se debe verificar que el entorno este levantado correctamente , que todos sus contenedores estén funcionando correctamente y tener replica set inicializado , sin olvidarse de que los datos deben estar importados.Con todo esto preparado entramos a la consola interactiva de Mongo.

Ingresamos a la Base de Datos

Agregamos los pipelines de forma directa

1.Empleado con salario más alto



2.Departamentos sin empleados asignados

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

RAM 2.51 GB CPU 11.34% Disk: 4.03 GB used (limit 1006.85 GB)

3. Sucursal que más ventas tiene por mes

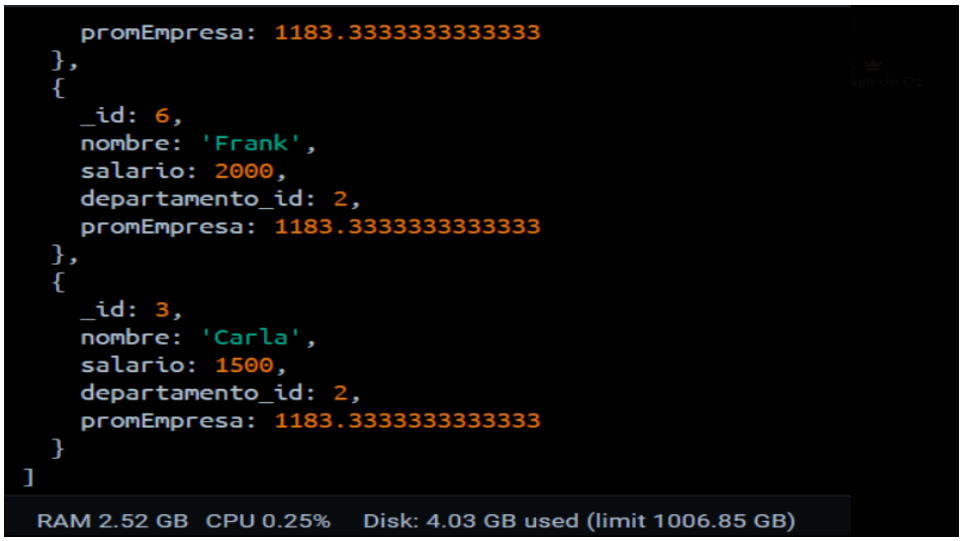
```
rs0 [direct: primary] escuela> db.ventas.aggregate([
...   { $group: { _id: { mes: "$_id.mes", sucursal: "$_id.sucursal" }, total: { $sum: "$total" } } },
...   { $sort: { "_id.mes": 1, total: -1 } },
...   { $group: { _id: "$_id.mes", topSucursal: { $first: "$_id.sucursal" }, maxTotal: { $first: "$total" } } }
... ])
...
[
  { _id: '2025-09', topSucursal: 'Quito', maxTotal: 39000 },
  { _id: '2025-08', topSucursal: 'Guayaquil', maxTotal: 42000 }
]
rs0 [direct: primary] escuela>
```

- Empleados con mayor salario promedio en la empresa
  (Con ventanas)

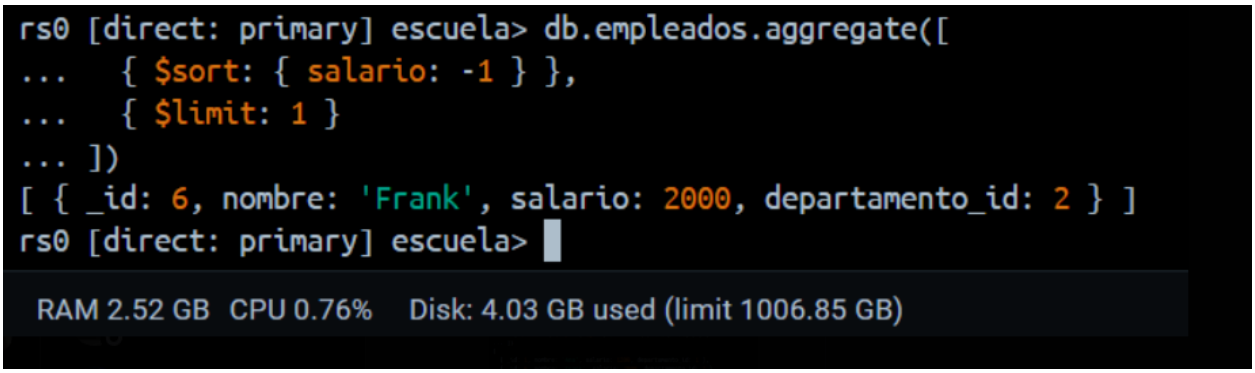
```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       partitionBy: null,
...       output: {
...         promEmpresa: { $avg: "$salario" }
...       }
...     }
...   },
...   { $match: { $expr: { $gt: ["$salario", "$promEmpresa"] } } }
... ])
[
  {
    _id: 1,
    nombre: 'Ana',
    salario: 1200,
    departamento_id: 1,
```

RAM 2.52 GB CPU 3.14% Disk: 4.03 GB used (limit 1006.85 GB)

- Misma consulta (Sin ventanas)



- Empleado con el salario más alto



- Mostrar el salario promedio de un departamento para cada empleado

```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       partitionBy: "$departamento_id",
...       output: {
...         promDep: { $avg: "$salario" }
...       }
...     }
...   },
...   { $project: { nombre: 1, salario: 1, departamento_id: 1, promDep: 1 } }
... ])
[
  {
    _id: 1,
    nombre: 'Ana',
    salario: 1200,
    departamento_id: 1,
RAM 2.52 GB  CPU 2.14%  Disk: 4.03 GB used (limit 1006.85 GB)
```

```
    departamento_id: 2,
    promDep: 1433.3333333333333
  },
  {
    _id: 3,
    nombre: 'Carla',
    salario: 1500,
    departamento_id: 2,
    promDep: 1433.3333333333333
  },
  {
    _id: 5,
    nombre: 'Elena',
    salario: 700,
    departamento_id: 3,
    promDep: 700
  }
```

- Departamentos cuyo promedio salarial es mayo al promedio general

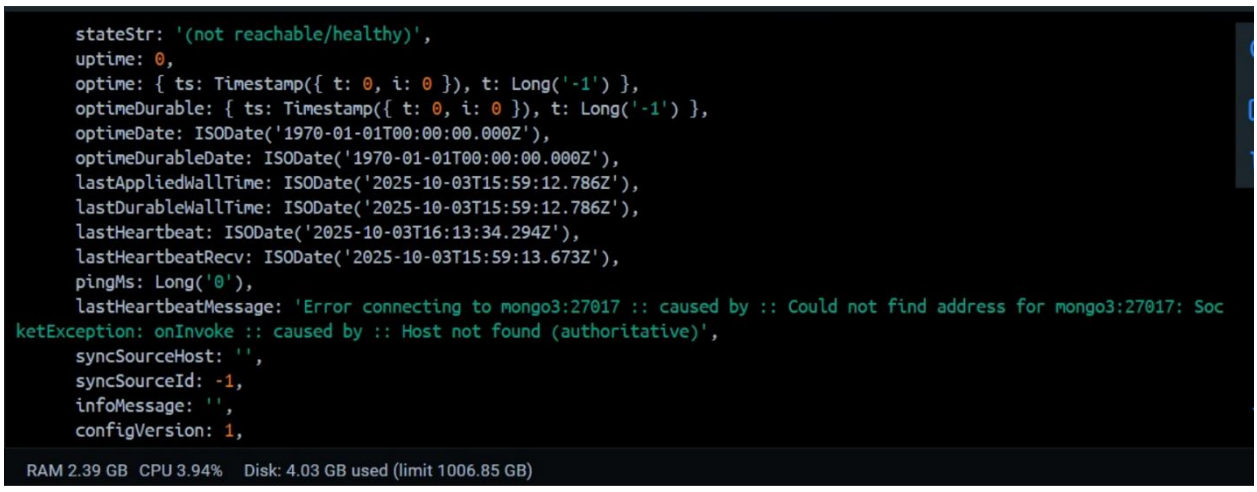
```
rs0 [direct: primary] escuela> db.empleados.aggregate([
...   {
...     $setWindowFields: {
...       partitionBy: "$departamento_id",
...       output: {
...         promDep: { $avg: "$salario" }
...       }
...     }
...   },
...   {
...     $setWindowFields: {
...       partitionBy: null,
...       output: {
...         promGlobal: { $avg: "$salario" }
...       }
...     }
...   },
```

- Departamentos cuyo promedio salarial es mayo al promedio general (Método sin ventanas)

PRUEBA DE RESILIENCIA

Detenemos un nodo

Con **rs.status** nos conectamos al REPLICA SET y verificamos que mongo3 está apagado



```
      configVersion: 1,
      configTerm: 2
    }
  ],
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1759508012, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1759508012, i: 1 })
}
rs0 [direct: secondary] escuela>
```

Ejecutamos el contenedor del cliente de MongoDB

```
Terminal
PS C:\Users\gatit\OneDrive\Escritorio\Practica_Docker> docker exec -it mongo1 mongosh
>>
Current Mongosh Log ID: 68dff05842070b2e79ce5f46

------
   The server generated these startup warnings when booting
   2025-10-03T15:47:50.408+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See
 http://dochub.mongodb.org/core/prodnotes-filesystem
   2025-10-03T15:47:52.164+00:00: Access control is not enabled for the database. Read and write access to data and config
uration is unrestricted
   2025-10-03T15:47:52.165+00:00: vm.max_map_count is too low
------

rs0 [direct: secondary] test>
```

Realizamos la consulta

```
rs0 [direct: secondary] escuela> db.empleados.find().limit(3)
[
  { _id: 5, nombre: 'Elena', salario: 700, departamento_id: 3 },
  { _id: 1, nombre: 'Ana', salario: 1200, departamento_id: 1 },
  { _id: 6, nombre: 'Frank', salario: 2000, departamento_id: 2 }
]
rs0 [direct: secondary] escuela>
RAM 2.45 GB  CPU 1.75%  Disk: 4.03 GB used (limit 1006.85 GB)
```

En este caso no sucederá nada ya que como el nodo primario esta activo las escrituras y lecturas continuaran sin ningún problema, únicamente habrá algún fallo si en una consulta se desea leer específicamente del nodo apagado o detenido.

De ser que el nodo primario se detenga, habrá un breve lapsus de pausa mientras uno de los nodos secundarios toma el rol, luego de esto el trabajo continuara con normalidad.

2.7 Resultados obtenidos

Durante la realización del taller se logró levantar correctamente el clúster de MongoDB utilizando Docker y configurar el Replica Set con los tres nodos. Se verificó que todos los contenedores estuvieran activos y sincronizados, y se pudo importar exitosamente la información desde los archivos .json a la base de datos.

Las consultas de agregación y cálculo de promedios se ejecutaron correctamente, mostrando resultados precisos como el empleado con mayor salario, los departamentos sin empleados asignados y la sucursal con más ventas. Además, la prueba de resiliencia demostró que el sistema podía continuar operando de manera normal incluso cuando uno de los nodos secundarios se detuvo, confirmando la eficacia de la replicación y la tolerancia a fallos del clúster.

2.8 Habilidades blandas empleadas en la práctica

☐ Liderazgo
☐ Trabajo en equipo
☐ Comunicación asertiva
☐ La empatía
☒ Pensamiento crítico
☐ Flexibilidad
☒ La resolución de conflictos
☒ Adaptabilidad
☒ Responsabilidad

Durante el desarrollo de la práctica se aplicaron diversas habilidades blandas que facilitaron la correcta implementación y el análisis de las consultas y la gestión del clúster de MongoDB.

El pensamiento crítico fue fundamental para planificar la configuración del Replica Set y analizar cómo se estructuraban las colecciones, definiendo la lógica adecuada para cada consulta de agregación, como calcular salarios promedio o identificar la sucursal con más ventas.

La resolución de conflictos resultó necesaria al enfrentar errores en la importación de datos, la sincronización entre nodos y la ejecución de pipelines complejos, ajustando comandos y operadores para asegurar que los resultados fueran correctos.

La adaptabilidad se puso en práctica al ajustar la estrategia de trabajo frente a situaciones imprevistas, como la detención de un nodo durante la prueba de resiliencia, demostrando flexibilidad ante problemas técnicos y cambios en el entorno.

La responsabilidad se reflejó en el cumplimiento de cada objetivo de la práctica, verificando que los datos importados y las consultas ejecutadas fueran correctos, garantizando la coherencia y calidad del trabajo realizado.

2.9 Conclusiones

- Se configuraron correctamente los contenedores y el Replica Set, asegurando la comunicación entre los nodos y su correcto estado.
- Se importaron y gestionaron los datos dentro del clúster, verificando que se sincronizaran correctamente y estuvieran disponibles para las consultas.
- Se evaluó la resiliencia del clúster mediante la detención de un nodo, comprobando que el sistema continuara funcionando sin interrupciones.

2.10 Recomendaciones

- Verificar siempre que todos los contenedores estén activos antes de ejecutar consultas para evitar errores de conexión o datos incompletos.
- Realizar pruebas periódicas de resiliencia deteniendo nodos de manera controlada, para garantizar que el clúster pueda manejar fallos sin afectar el funcionamiento del sistema.

2.11 Anexos

Todos los documentos obtenidos durante la practica se encuentran subidos en el bloc de notas (ONE NOTE).