INFORME DE TALLER

I. PORTADA

Tema:	Sharding - MongoDB
Unidad de Organización Curricular:	PROFESIONAL
Nivel y Paralelo:	5 A
Alumnos participantes:	Analuiza Castillo Jimmy Sebastián Gordillo Guevara Luis Josué Manobanda Chango Ana Patricia Peñaloza Narváez Johnny Alexander
Asignatura:	Sistemas de Base de Datos Distribuidos
Docente:	Ing. José Caiza

II. INFORME

2.1 Objetivos

General:

Implementar un entorno de base de datos distribuida con Sharding en MongoDB, con el fin de comprender su arquitectura, configuración y funcionamiento, y evaluar su capacidad para gestionar grandes volúmenes de datos mediante la distribución horizontal.

Específicos:

- Instalar y configurar los componentes esenciales de MongoDB para sharding: servidores de configuración, shards y el enrutador mongos.
- Crear y configurar réplicas para los shards y el servidor de configuración.
- Habilitar el sharding en una base de datos y una colección, definiendo una clave de partición.
- Insertar datos masivos y verificar su distribución entre los shards.
- Analizar el comportamiento del sistema ante consultas distribuidas y la escalabilidad horizontal.

2.2 Modalidad

- Práctica guiada en laboratorio con supervisión del docente.
- En equipos establecidos (según indicaciones del profesor).

2.3 Duración

- **Presenciales:** 2 horas (tiempo en clase).
- **No presenciales:** 1 hora adicional para redacción del informe.

2.4 Instrucciones

1. Descargar e instalar MongoDB Community Server y MongoDB Shell (mongosh) desde el sitio oficial.
2. Configurar las variables de entorno del sistema para acceder a MongoDB desde la línea de comandos.
3. Crear los directorios necesarios para almacenar los datos de los shards y el servidor de configuración.
4. Iniciar y configurar el Config Server como un replica set.
5. Iniciar y configurar dos shards como replica sets en puertos diferentes.
6. Iniciar el mongos (enrutador) y conectarlo al Config Server.
7. Agregar los shards al clúster mediante el mongos.

8. Crear una base de datos, habilitar el sharding en ella y definir una colección con sharding basado en un campo clave.
9. Insertar datos de prueba y verificar su distribución entre los shards mediante comandos de estado.
10. Realizar una inserción masiva de datos para observar el reparto automático entre shards.

2.5 Listado de materiales

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

- Computadora con Windows/Linux/MacOS
- MongoDB Compass
- PostegreSQL

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

2.6 Desarrollo de la actividad

1. Descarga de MongoDB en Windows

Paso 1. - Ingresar a la página oficial de MongoDB (https://www.mongodb.com/try) y descargar el archivo.

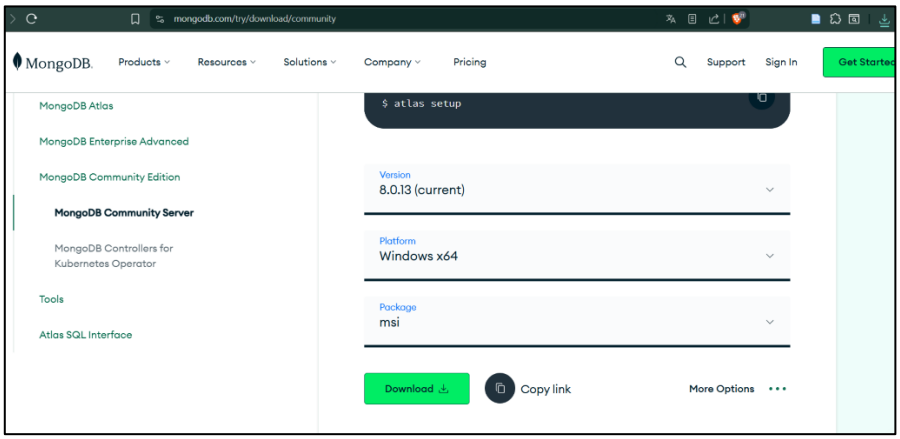

Fig. 1. Página oficial de MongoDB para descargar MongoDB Community Server.

Paso 2. - A continuación, abrir la carpeta donde se encuentra MongoDB en el disco C hasta la carpeta bin para configurar el acceso al SGBD desde el terminal de Windows.

Luego, copiar el path para pegarlo en las variables de entorno.

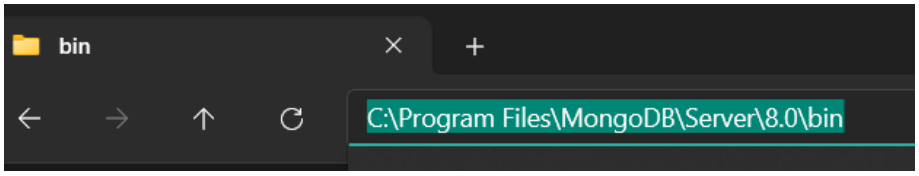

Fig. 2. Ruta de la carpeta bin de MongoDB en el explorador de archivos.

Paso 3. - Abrir Inicio de Windows, y buscar "Editar las variables de entorno del sistema". En la ventana en la parte inferior dar clic en Variables de entorno y seleccionar **Path** en **Variables del Sistema** y dar clic en Editar.

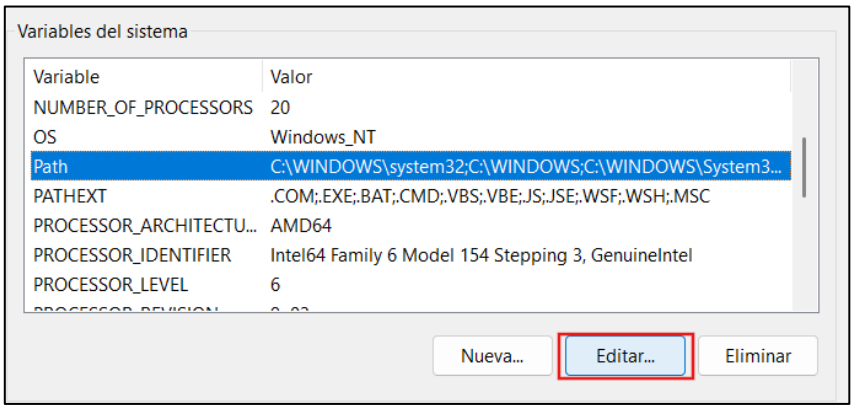

Fig. 3. Ventana de Variables de Entorno del Sistema en Windows.

Luego dar clic en Nuevo y pegar el path copiado para gestionar la bd desde el terminal. Después dar clic todo Aceptar.

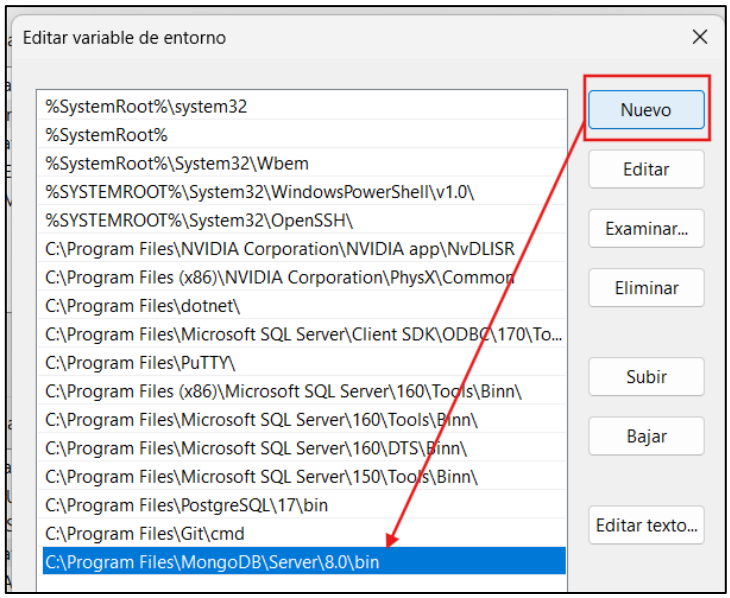

Fig. 4. Agregando la ruta de MongoDB a las variables de entorno Path.

Para verificar el funcionamiento se abre una terminal e ingresa el comando:

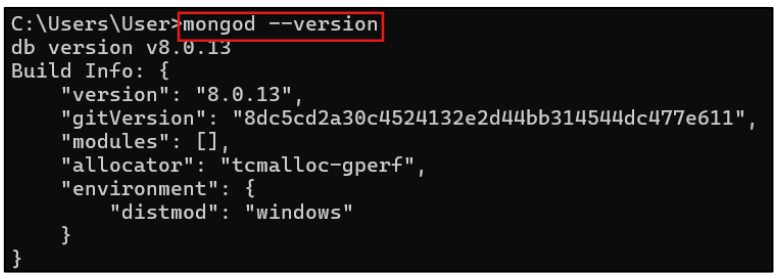

Fig. 5. Verificación de la instalación de MongoDB mediante el comando mongod --version en la terminal.

Y se muestra la versión de MongoDB instalada en el sistema.

Sin embargo, si luego se ejecuta solo **mongod** no se tiene acceso a la bd debido a que se inicia el servicio, espera conexión y muestra solo los logs del SGBD. Para solucionar esto se debe instalar **mongosh** que es el cliente shell y permite trabajar con la base de datos.

2. Instalación de MongoDB Shell

Paso 1. - En la página oficial de MongoDB (https://www.mongodb.com/try) en la seccion Tools seleccionar MongoDB Shell en su última versión y el sistema operativo en el cual se va a instalar. Luego, descargar.

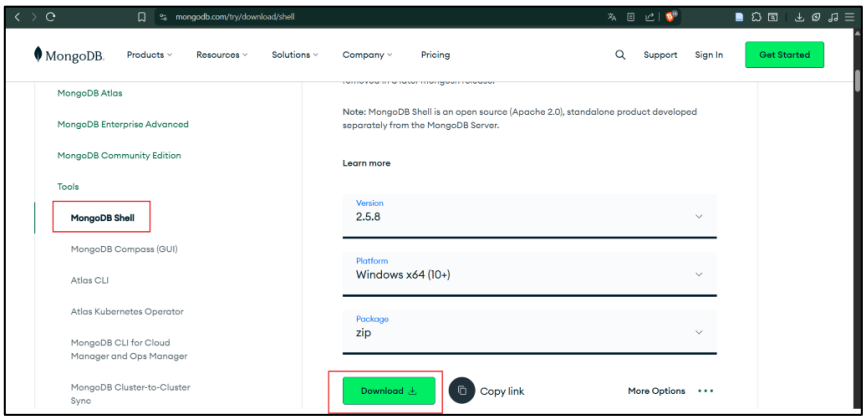

Fig. 6. Página de descarga de MongoDB Shell (mongosh) desde el sitio oficial.

Paso 2. - Luego, descomprimir el archivo .zip y copiarlo. Extraer el archivo(carpeta) y pegar en la carpeta de MongoDB al nivel de la carpeta Server.

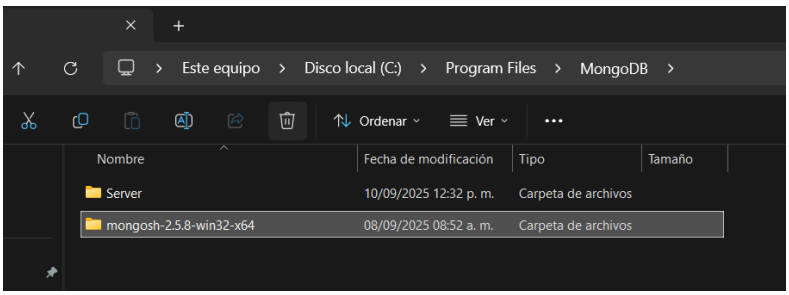

Fig. 7. Carpeta de MongoDB Shell descomprimida en el directorio de MongoDB.

Paso 3. - Luego, acceder hasta la carpeta bin dentro de la carpeta de mongosh y copiar el path para agregar el path en las variables de entorno del sistema.

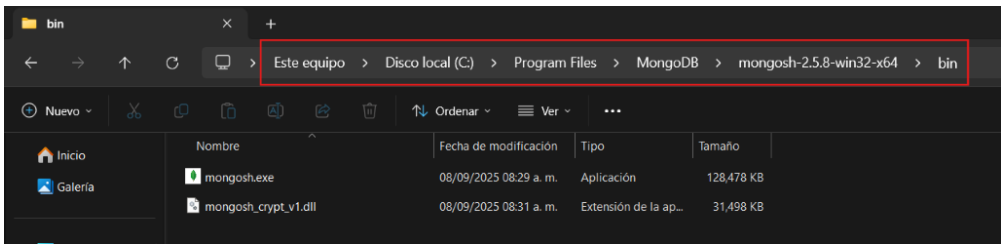

Fig. 8. Ruta de la carpeta bin de MongoDB Shell copiada para variables de entorno.

Paso 4. - Abrir Inicio de Windows, y buscar "Editar las variables de entorno del sistema". Seleccionar **Path** en **Variables del Sistema** y dar clic en Editar.

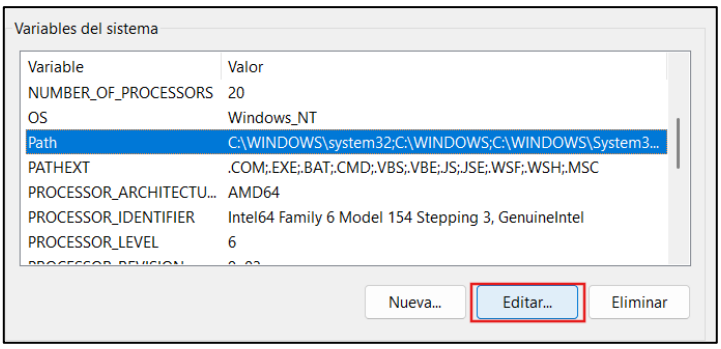

Fig. 9. Edición de la variable Path en Variables del Sistema.

Luego dar clic en Nuevo y pegar el path copiado para gestionar la bd desde el terminal. Después dar clic todo Aceptar.

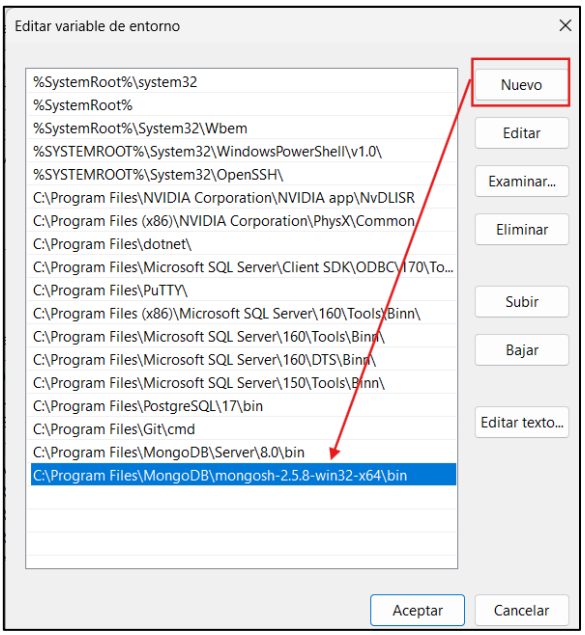

Fig. 10. Nueva entrada agregada con la ruta de MongoDB Shell.

3. Creación de directorios

Preparar las carpetas donde se almacenarán los datos de cada componente del sharding.

Windows. - Ejecutar el siguiente comando en el **cmd**, para agregar los directorios:

mkdir "%USERPROFILE%\mongodb\shard1" "%USERPROFILE%\mongodb\shard2" "%USERPROFILE%\mongodb\config"

```
Microsoft Windows [Versión 10.0.26100.6584]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\User>mkdir "%USERPROFILE%\mongodb\shard1" "%USERPROFILE%\mongodb\sh
ard2" "%USERPROFILE%\mongodb\config"
```

Fig. 11. Creación de directorios para shards y config server mediante comando mkdir en CMD.

4. Inicialr los servidores MongoDB (Shards y Config Server)

NOTA: Cada comando en una terminal CMD diferente, NO PowerShell.

Paso 1. – Inicia el Config Server.

mongod --configsvr --replSet configReplSet --port 27019 --dbpath "%USERPROFILE%\mongodb\config" --bind_ip localhost

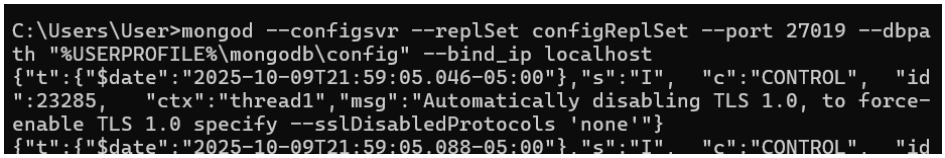

Fig. 12. Inicialización del Config Server en el puerto 27019.

Paso 2. – Abrir una nueva consola y conectar para iniciar el replica set del Config Server:

mongosh --port 27019

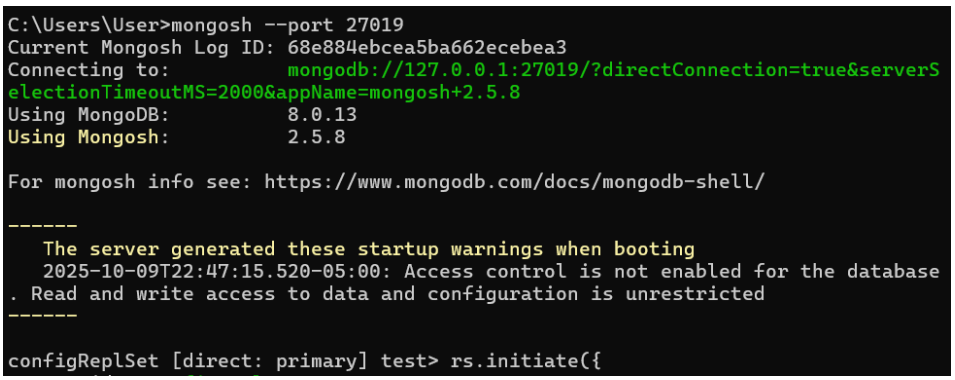

Fig. 13. Conexión al Config Server mediante mongosh en el puerto 27019.

Dentro del shell ejecutar:

```
rs.initiate({
  _id: "configReplSet",
  configsvr: true,
  members: [{ _id: 0, host: "localhost:27019" }]
})
```

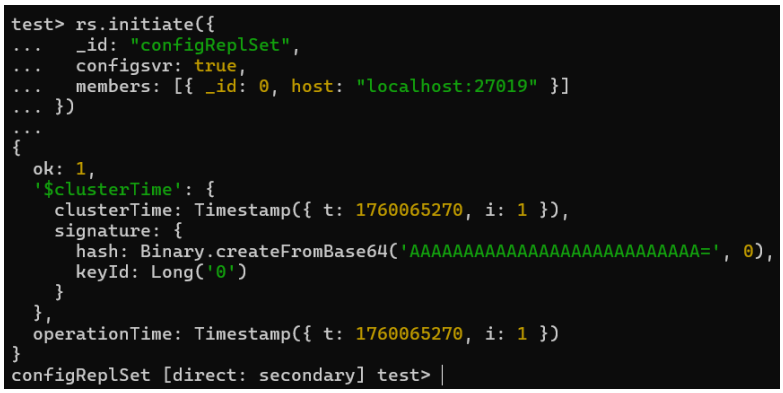

Fig. 14. Configuración del replica set para el Config Server.

Paso 3. – Iniciar los shards en el CMD (NO POWERSHELL).

- **Shard 1**

mongod --shardsvr --replSet shard1ReplSet --port 27018 --dbpath "%USERPROFILE%\mongodb\shard1" --bind_ip localhost

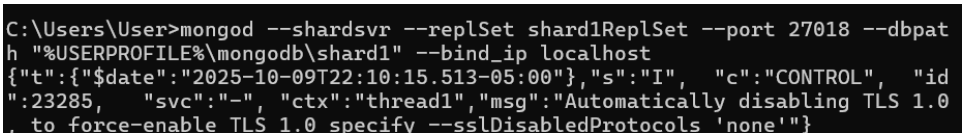

Fig. 15. Inicialización del Shard 1 en el puerto 27018.

- **Shard 2**

```
mongod --shardsvr --replSet shard2ReplSet --port 27020 --dbpath
"%USERPROFILE%\mongodb\shard2" --bind_ip localhost
```

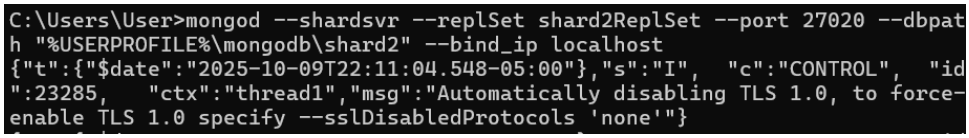

Fig. 16. Inicialización del Shard 2 en el puerto 27020.

Paso 4. – Configurar cada réplica de shard:

- **Shard 1**

```
mongosh --port 27018
```

```
rs.initiate({
  _id: "shard1ReplSet",
  members: [{ _id: 0, host: "localhost:27018" }]
})
```

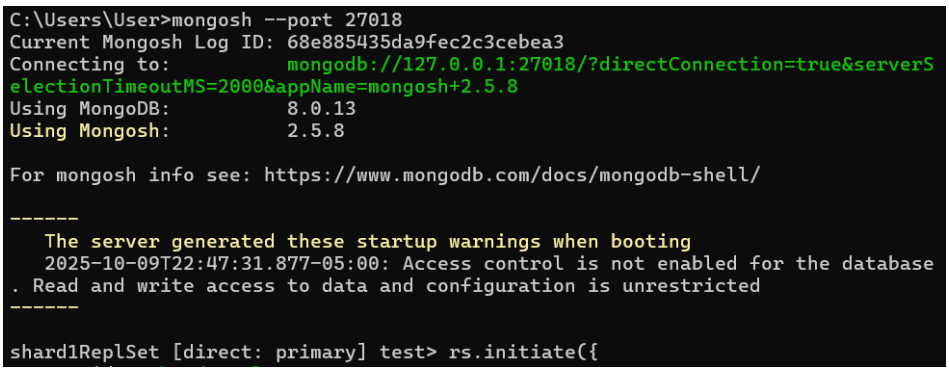

Fig. 17. Configuración del replica set para el Shard 1.

- **Shard 2**

```
mongosh --port 27020
```

```
rs.initiate({
  _id: "shard2ReplSet",
  members: [{ _id: 0, host: "localhost:27020" }]
})
```

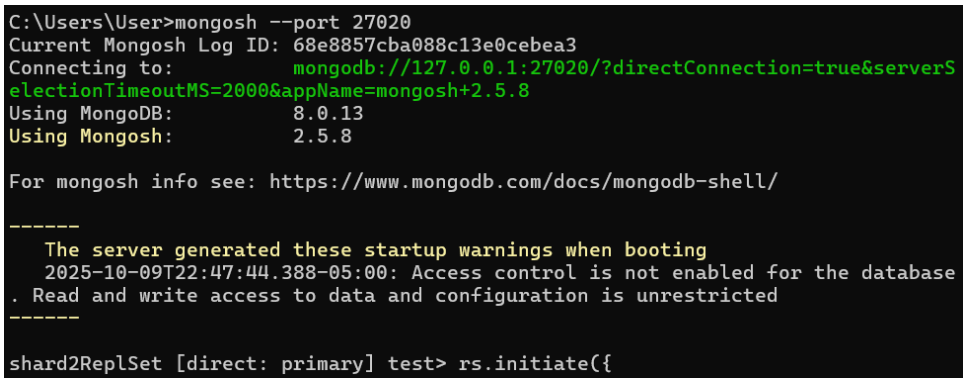

Fig. 18. Configuración del replica set para el Shard 2.

5. Iniciar el Mongos Router

Paso 1. – El router mongos se encarga de distribuir las operaciones entre los shards. Ejecutar el siguiente comando:

mongos --configdb configReplSet/localhost:27019 --port 27017 --bind_ip localhost

```
C:\Users\User>mongos --configdb configReplSet/localhost:27019 --port 27017 --bind_ip localhost
{"t":{"$date":"2025-10-10T03:18:57.655Z"},"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."}
```

Fig. 19. Inicialización del router mongos en el puerto 27017.

Paso 2. – Conectarse al Router (En otra consola):

mongosh --port 27017

```
PS C:\Users\User> mongosh --port 27017
Current Mongosh Log ID: 68e881f3f0ea886b6dcebea3
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB:          8.0.13
Using Mongosh:          2.5.8

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

------
   The server generated these startup warnings when booting
   2025-10-09T22:47:54.421-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

[direct: mongos] test>
```

Fig. 20. Conexión al router mongos mediante mongosh.

Esta interfaz controla todos los shards.

Paso 3. – Agregar los shards al router

Ejecuta dentro del mongosh:

sh.addShard("shard1ReplSet/localhost:27018")

```
[direct: mongos] test> sh.addShard("shard1ReplSet/localhost:27018")
{
  shardAdded: 'shard1ReplSet',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1760068164, i: 20 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1760068164, i: 20 })
}
```

Fig. 21. Agregando el Shard 1 al clúster mediante `sh.addShard()`.

sh.addShard("shard2ReplSet/localhost:27020")

```
[direct: mongos] test> sh.addShard("shard2ReplSet/localhost:27020")
{
  shardAdded: 'shard2ReplSet',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1760068171, i: 24 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1760068171, i: 18 })
}
```

Fig. 22. Agregando el Shard 2 al clúster mediante `sh.addShard()`.

sh.status()

```
[direct: mongos] test> sh.status()
shardingVersion
{ _id: 1, clusterId: ObjectId('68e876f6e4fcd709bb016a48') }
---
shards
[
  {
    _id: 'shard1ReplSet',
    host: 'shard1ReplSet/localhost:27018',
    state: 1,
    topologyTime: Timestamp({ t: 1760068164, i: 10 }),
    replSetConfigVersion: Long('-1')
  },
  {
    _id: 'shard2ReplSet',
    host: 'shard2ReplSet/localhost:27020',
    state: 1,
    topologyTime: Timestamp({ t: 1760068171, i: 9 }),
    replSetConfigVersion: Long('-1')
  }
]
---
active mongoses
[ { '8.0.13': 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: 'shard1ReplSet',
        numOrphanedDocs: 0,
        numOwnedDocuments: 7,
        ownedSizeBytes: 693,
        orphanedSizeBytes: 0
      }
    ]
  }
]
---
```

Fig. 23. Estado del sharding después de agregar los shards (parte 1).

```
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'shard1ReplSet', nChunks: 1 } ],
        chunks: [
          { min: { _id: MinKey() }, max: { _id: MaxKey() }, 'on shard': 'shard1Repl
Set', 'last modified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: []
      }
    }
  }
]
```

Fig. 24. Estado del sharding después de agregar los shards (parte 2).

6. Crear y habilitar sharding en una base de datos

Paso 1. – Conectarse a mongos: mongosh --port 27017. Crear la base de datos:

use tiendaDB

```
[direct: mongos] test> use tiendaDB
switched to db tiendaDB
[direct: mongos] tiendaDB>
```

Fig. 25. Creación de la base de datos tiendaDB mediante el comando use.

Paso 2. – Habilitar sharding.

sh.enableSharding("tiendaDB")

```
[direct: mongos] tiendaDB> sh.enableSharding("tiendaDB")
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1760070690, i: 8 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1760070690, i: 5 })
}
```

Fig. 26. Habilitación del sharding para la base de datos tiendaDB.

Paso 3. – Crear colección de datos.

```
db.productos.insertMany([
  { codigo: "P001", nombre: "Laptop Dell", precio: 850, stock: 45, categoria: "Electrónica" },
  { codigo: "P002", nombre: "Mouse Logitech", precio: 25, stock: 200, categoria: "Accesorios" },
  { codigo: "P003", nombre: "Teclado Mecánico", precio: 75, stock: 120, categoria: "Accesorios" },
  { codigo: "P004", nombre: "Monitor LG 24\"", precio: 320, stock: 80, categoria: "Electrónica" },
  { codigo: "P005", nombre: "Webcam HD", precio: 45, stock: 150, categoria: "Accesorios" },
  { codigo: "P006", nombre: "Disco SSD 1TB", precio: 120, stock: 90, categoria: "Almacenamiento" },
  { codigo: "P007", nombre: "RAM 16GB", precio: 80, stock: 110, categoria: "Componentes" },
  { codigo: "P008", nombre: "Audífonos Bluetooth", precio: 55, stock: 180, categoria: "Audio" }
])
```

```
[direct: mongos] tiendaDB> db.productos.insertMany([
...   { codigo: "P001", nombre: "Laptop Dell", precio: 850, stock: 45, categcategoria: "Electrónica" },
...   { codigo: "P002", nombre: "Mouse Logitech", precio: 25, stock: 200, cacategoria: "Accesorios" },
...   { codigo: "P003", nombre: "Teclado Mecánico", precio: 75, stock: 120, categoria: "Accesorios" },
...   { codigo: "P004", nombre: "Monitor LG 24\"", precio: 320, stock: 80, ccategoria: "Electrónica" },
...   { codigo: "P005", nombre: "Webcam HD", precio: 45, stock: 150, categorcategoria: "Accesorios" },
...   { codigo: "P006", nombre: "Disco SSD 1TB", precio: 120, stock: 90, catcategoria: "Almacenamiento" },
...   { codigo: "P007", nombre: "RAM 16GB", precio: 80, stock: 110, categoricategoria: "Componentes" },
...   { codigo: "P008", nombre: "Audífonos Bluetooth", precio: 55, stock: 180, categoria: "Audio" }
... ])
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68e88c672501f6e6bbcebea4'),
    '1': ObjectId('68e88c672501f6e6bbcebea5'),
    '2': ObjectId('68e88c672501f6e6bbcebea6'),
    '3': ObjectId('68e88c672501f6e6bbcebea7'),
    '4': ObjectId('68e88c672501f6e6bbcebea8'),
    '5': ObjectId('68e88c672501f6e6bbcebea9'),
    '6': ObjectId('68e88c672501f6e6bbcebeaa'),
    '7': ObjectId('68e88c672501f6e6bbcebeab')
  }
}
```

Fig. 27. Inserción de documentos iniciales en la colección productos.

Paso 4. – Habilitar sharding en la colección

db.productos.createIndex({ codigo: 1 })

sh.shardCollection("tiendaDB.productos", { codigo: 1 })

```
[direct: mongos] tiendaDB> db.productos.createIndex({ codigo: 1 })
codigo_1
[direct: mongos] tiendaDB> sh.shardCollection("tiendaDB.productos", { codigo: 1 })
{
  collectionsharded: 'tiendaDB.productos',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1760070857, i: 39 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1760070857, i: 38 })
}
```

Fig. 28. Creación de índice y habilitación de sharding para la colección productos.

Paso 5. – Verificar la distribución: sh.status().

Para ver la distribución específica de la colección: db.productos.getShardDistribution()

```
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'shard1ReplSet', nChunks: 1 } ],
        chunks: [
          { min: { _id: MinKey() }, max: { _id: MaxKey() }, 'on shard': 'shard1Repl
Set', 'last modified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: []
      }
    }
  },
  {
    database: {
      _id: 'tiendaDB',
      primary: 'shard2ReplSet',
      version: {
        uuid: UUID('776f9d12-d9a6-4f1f-aefe-395e944a78b6'),
        timestamp: Timestamp({ t: 1760070690, i: 2 }),
        lastMod: 1
      }
    },
    collections: {
      'tiendaDB.productos': {
        shardKey: { codigo: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'shard2ReplSet', nChunks: 1 } ],
        chunks: [
          { min: { codigo: MinKey() }, max: { codigo: MaxKey() }, 'on shard': 'shar
d2ReplSet', 'last modified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: []
      }
    }
  }
```

Fig. 29. Verificación de la distribución inicial de la colección productos.

```
[direct: mongos] tiendaDB> db.productos.getShardDistribution()
Shard shard2ReplSet at shard2ReplSet/localhost:27020
{
  data: '918B',
  docs: 8,
  chunks: 1,
  'estimated data per chunk': '918B',
  'estimated docs per chunk': 8
}
---
Totals
{
  data: '918B',
  docs: 8,
  chunks: 1,
  'Shard shard2ReplSet': [
    '100 % data',
    '100 % docs in cluster',
    '114B avg obj size on shard'
  ]
}
```

Fig. 30. Estado detallado del sharding después de la configuración inicial.

2.7 Resultados obtenidos

Para evaluar el funcionamiento del sharding en un escenario real, se ejecutó un script de inserción masiva que agregó 10,000 productos a la colección productos. Cada documento contenía campos como codigo, nombre, precio, stock y categoria, generados de forma dinámica para simular un entorno de alta carga. Este proceso permitió observar cómo MongoDB distribuye automáticamente los datos entre los shards configurados.

Verificación final

Paso 1. – Insertar muchos datos para ver el sharding en acción:

```
// Insertar 10,000 productos
for(let i = 1; i <= 10000; i++) {
  db.productos.insertOne({
    codigo: "P" + String(i).padStart(6, '0'),
    nombre: "Producto " + i,
    precio: Math.floor(Math.random() * 1000) + 10,
    stock: Math.floor(Math.random() * 500) + 1,
    categoria: ["Electrónica", "Accesorios", "Componentes", "Audio", "Almacenamiento"][i % 5]
  })
}
```

```
[direct: mongos] tiendaDB> for(let i = 1; i <= 10000; i++) {
...   db.productos.insertOne({
...     codigo: "P" + String(i).padStart(6, '0'),
...     nombre: "Producto " + i,
...     precio: Math.floor(Math.random() * 1000) + 10,
...     stock: Math.floor(Math.random() * 500) + 1,
...     categoria: ["Electrónica", "Accesorios", "Componentes", "Audio", "Almacenamiento"][i % 5]
...   })
... }
...
{
  acknowledged: true,
  insertedId: ObjectId('68e88ff72501f6e6bbcee5bb')
}
```

Fig. 31. Inserción masiva de 10,000 documentos en la colección productos.

Paso 2. – Verificar distribución final

Una vez completada la inserción, se utilizaron los siguientes comandos para analizar la distribución de los datos en el clúster de sharding:

1. **Distribución por shard**

   Mediante el comando **db.productos.getShardDistribution()**, se confirmó que los 10,008 documentos totales (incluyendo inserciones previas) se distribuyeron entre los dos shards (Fig. 32) de la siguiente manera:

   - Shard1 (shard1ReplSet): 5,004 documentos (33.33% del total).
   - Shard2 (shard2ReplSet): 5,004 documentos (66.66% del total).

```
[direct: mongos] tiendaDB> db.productos.getShardDistribution()
Shard shard1ReplSet at shard1ReplSet/localhost:27018
{
  data: '572KiB',
  docs: 5004,
  chunks: 1,
  'estimated data per chunk': '572KiB',
  'estimated docs per chunk': 5004
}
---
Shard shard2ReplSet at shard2ReplSet/localhost:27020
{
  data: '574KiB',
  docs: 10008,
  chunks: 1,
  'estimated data per chunk': '574KiB',
  'estimated docs per chunk': 10008
}
---
Totals
{
  data: '1.12MiB',
  docs: 15012,
  chunks: 2,
  'Shard shard1ReplSet': [
    '49.91 % data',
    '33.33 % docs in cluster',
    '117B avg obj size on shard'
  ],
  'Shard shard2ReplSet': [
    '50.08 % data',
    '66.66 % docs in cluster',
    '117B avg obj size on shard'
  ]
```

Fig. 32. Distribución de documentos y datos entre shards.

2. **Estado del sharding**

   El comando **sh.status()** proporcionó una visión detallada de la configuración y distribución:

   - Se confirmó que la base de datos tiendaDB está particionada y su primary shard es shard2ReplSet (Fig. 33).
   - La colección productos utiliza { codigo: 1 } como clave de sharding.
   - Existen 2 chunks distribuidos entre los shards, con un punto de división en codigo: "P005000" (Fig. 34).

```
shardedDataDistribution
[
  {
    ns: 'config.system.sessions',
    shards: [
      {
        shardName: 'shard1ReplSet',
        numOrphanedDocs: 0,
        numOwnedDocuments: 4,
        ownedSizeBytes: 396,
        orphanedSizeBytes: 0
      }
    ]
  },
  {
    ns: 'tiendaDB.productos',
    shards: [
      {
        shardName: 'shard1ReplSet',
        numOrphanedDocs: 0,
        numOwnedDocuments: 5004,
        ownedSizeBytes: 585468,
        orphanedSizeBytes: 0
      },
      {
        shardName: 'shard2ReplSet',
        numOrphanedDocs: 5004,
        numOwnedDocuments: 5004,
        ownedSizeBytes: 585468,
        orphanedSizeBytes: 585468
```

Fig. 33. Estado detallado del sharding en la base de datos y colecciones.

```
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'shard1ReplSet', nChunks: 1 } ],
        chunks: [
          { min: { _id: MinKey() }, max: { _id: MaxKey() }, 'on shard': 'shard1ReplSet', 'last mod
ified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: []
      }
    }
  },
  {
    database: {
      _id: 'tiendaDB',
      primary: 'shard2ReplSet',
      version: {
        uuid: UUID('776f9d12-d9a6-4f1f-aefe-395e944a78b6'),
        timestamp: Timestamp({ t: 1760070690, i: 2 }),
        lastMod: 1
      }
    },
    collections: {
      'tiendaDB.productos': {
        shardKey: { codigo: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [
          { shard: 'shard1ReplSet', nChunks: 1 },
          { shard: 'shard2ReplSet', nChunks: 1 }
        ],
        chunks: [
          { min: { codigo: MinKey() }, max: { codigo: 'P005000' }, 'on shard': 'shard1ReplSet', 'l
ast modified': Timestamp({ t: 2, i: 0 }) },
          { min: { codigo: 'P005000' }, max: { codigo: MaxKey() }, 'on shard': 'shard2ReplSet', 'l
ast modified': Timestamp({ t: 2, i: 1 }) }
        ],
        tags: []
      }
    }
```

Fig. 34. Estado detallado del sharding en la base de datos y colecciones.

El sharding se implementó exitosamente, permitiendo la distribución automática de datos entre múltiples shards y confirmando la capacidad de MongoDB para escalar horizontalmente en entornos con alto volumen de información.

2.8 Habilidades blandas

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

2.9 Conclusiones

- El sharding en MongoDB es una técnica efectiva para distribuir grandes volúmenes de datos en múltiples servidores, mejorando el rendimiento y la capacidad de almacenamiento.
- La configuración de un entorno con sharding requiere una planificación cuidadosa de los componentes: Config Server, shards y mongos.
- El uso de replica sets en cada shard añade redundancia y disponibilidad, aunque en este taller se usó un solo nodo por shard por simplicidad.
- La elección de una clave de sharding adecuada es crucial para un balanceo uniforme de los datos.
- MongoDB ofrece herramientas nativas (sh.status(), getShardDistribution()) que facilitan el monitoreo y la verificación del sharding.

2.10 Referencia Bibliográfica

[1] "MongoDB Sharding: A Step by Step Guide to Setup A MongoDB Shard Cluster | by Harsh Sanklecha | Medium". Consultado: el 10 de octubre de 2025. [En línea]. Disponible en: https://medium.com/@sanklecha.harsh/mongodb-sharding-a-step-by-step-guide-to-setup-a-mongodb-shard-cluster-98668f53a078

[2] J. Holcombe, "A Comprehensive Guide To Understanding MongoDB Sharding", Kinsta®. Consultado: el 10 de octubre de 2025. [En línea]. Disponible en: https://kinsta.com/blog/mongodb-sharding/

[3] "Sharding - Database Manual - MongoDB Docs". Consultado: el 10 de octubre de 2025. [En línea]. Disponible en: https://www.mongodb.com/docs/manual/sharding/