



INFORME DE TALLER

I. PORTADA

Tema:	Taller de Sharding
Unidad de Organización Curricular:	PROFESIONAL
Nivel y Paralelo:	Quinto - A
Alumnos participantes:	Cholota Guamán Carlos Sebastián Mazabanda Pilamunga Diego Abraham Tixilema Puaquiza Kevin Alexander Tubon Chipantiza Danilo Alexander
Asignatura:	Sistemas de Bases de Datos Distribuidos
Docente:	Ing. José Rubén Caiza.

II. INFORME DE TALLER

2.1 Objetivos

General:

Implementar un sistema de base de datos distribuida aplicando el concepto de sharding en MongoDB y la replicación en PostgreSQL, para comprender su funcionamiento y ventajas.

Específicos:

- Configurar los entornos de MongoDB y PostgreSQL.
- Crear estructuras de datos distribuidas utilizando fragmentos (shards).
- Aplicar comandos de replicación y verificación.
- Analizar los resultados del proceso de fragmentación y replicación.

2.2 Modalidad

Presencial

2.3 Tiempo de duración

Presenciales: 11

No presenciales: 0

2.3.1 Instrucciones

- Descargar e instalar MongoDB desde el sitio oficial:
- Configurar las variables de entorno para poder ejecutar mongo desde la consola.
- Crear los directorios para los fragmentos (shards):
- Iniciar los servidores shard y config de MongoDB desde cada carpeta creada.
- Conectar el router (mongos) y enlazar los shards al clúster.
- Activar el sharding en la base de datos con el comando:
- Definir la colección a fragmentar:
- Insertar y consultar datos para comprobar la distribución entre los shards.
- Configurar la replicación en PostgreSQL, creando las tablas (VEHICULO, CONDUCTORES, etc.) y aplicando triggers de control.
- Verificar resultados, comprobando que los datos se distribuyen y replican correctamente entre los nodos.



2.4 Listado de equipos, materiales y recursos

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

- **Computadora.**
- **Aula virtual/internet**

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): _____

2.5 Actividades por desarrollar

Profundizar en la configuración avanzada del sharding y la replicación en entornos distribuidos. Se implementará la conexión entre múltiples servidores para evaluar el rendimiento y la tolerancia a fallos del sistema. Además, se realizará la integración entre MongoDB y PostgreSQL para analizar la interoperabilidad entre bases de datos NoSQL y relacionales.

2.6 Resultados obtenidos

El sharding es una técnica de particionado horizontal que permite escalar bases de datos NoSQL dividiendo colecciones en fragmentos distribuidos entre varios servidores. Este taller práctico muestra la configuración local de un clúster shardeado en MongoDB y ejemplos de replicación/validaciones en PostgreSQL para comprender tolerancia a fallos, escalabilidad y consistencia.

2.6.1 Descargar MongoDB Community Server

- Nos dirigimos a: <https://www.mongodb.com/try/download/community>.
- Seleccionar:
 - Version: 8.0.6 (current)
 - Platform: Windows x64
 - Package: msi

VersionVersión	8.0.6 (current)8.0.6 (actual)	▼
PlatformPlataforma	Windows x64Ventanas x64	▼
PackagePaquete	msi	▼

Download Descargar

Copy linkCopiar enlace

More OptionsMás opciones

...

Ilustración 1.Descarga de mongoDB



2.6.2 Descargar MONGODB SHELL

- Descargar MongoDB Shell (mongosh) desde la misma página; descomprimir e instalar.
- En la misma pagina de MongoDB, descargar mongosh
- Seleccionar:
 - Version: 2.5.8
 - Platform: Windows x64 (10+)
 - Package: zip

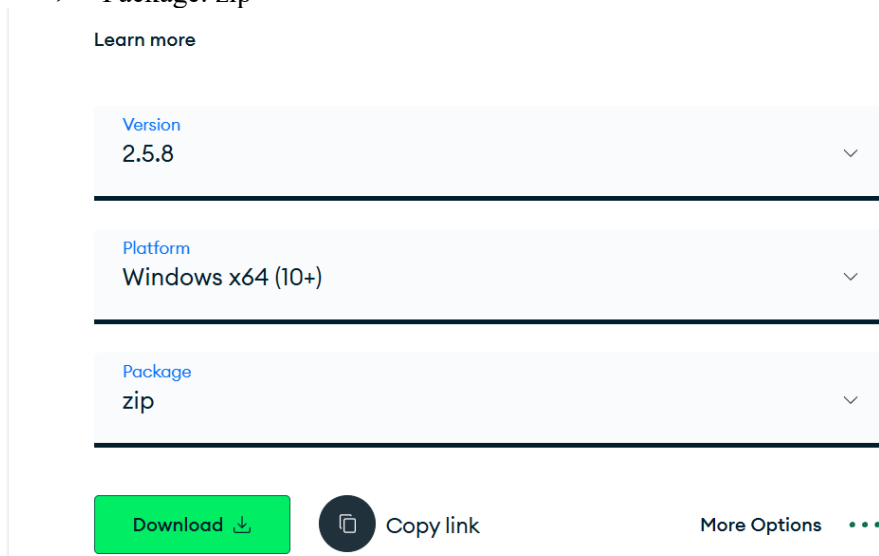


Ilustración 2. Descarga de MongoDB Shell (mongosh)

- La ruta es: C:\Users\ASUS\AppData\Local\Programs\mongosh\mongosh.exe

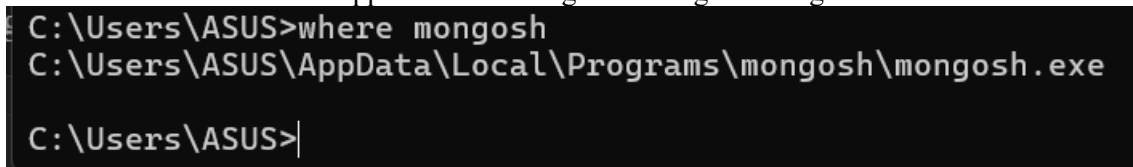


Ilustración 3. Verificación de ruta

2.6.3 Configurar las variables de entorno

- Para poder usar MongoDB desde cualquier ubicación en la terminal: presionar *Windows + R* > *sysdm.cpl*, elegir la pestaña "Opciones avanzadas" > "Variables de entorno"
- En "Variables del sistema", busca "Path" y "Editar"
- Click en "Nuevo" y agregar estas rutas:
C:\Program Files\MongoDB\Server\8.0\bin
C:\Program Files\MongoDB\mongosh\bin



- Verificar que se hayan creado

```
C:\Users\ASUS>dir "%USERPROFILE%\mongodb"
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 6C41-12C8

Directorio de C:\Users\ASUS\mongodb

12/10/2025  20:31    <DIR>          .
12/10/2025  20:31    <DIR>          ..
12/10/2025  20:31    <DIR>          config
12/10/2025  20:31    <DIR>          shard1
12/10/2025  20:31    <DIR>          shard2
               0 archivos                0 bytes
               5 dirs 208.412.614.656 bytes libres

C:\Users\ASUS>
```

Ilustración 7. Verificación de creación de directorio

2.6.5 Iniciar el Config Server

Al iniciar el Config Server, aparecen warnings normales indicando que ciertas colecciones aún no existen. Estos mensajes son esperados ya que el servidor aún no ha sido inicializado como parte del Replica Set.

- Abrir una ventana de cmd y ejecutar:

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

```
Símbolo del sistema - mongo
611, "svc": "S", "ctx": "initandlisten", "msg": "MongoDB starting", "attr": {"pid": 20628, "port": 27019, "dbPath": "C:/Users/ASUS/mongodb/config", "architecture": "64-bit", "host": "DESKTOP-MG3DUAB"}}
{"t": {"$date": "2025-10-12T20:37:53.257-05:00"}, "s": "I", "c": "CONTROL", "id": 23398, "svc": "S", "ctx": "initandlisten", "msg": "Target operating system minimum version", "attr": {"targetMinOS": "Windows 7/Windows Server 2008 R2"}}
{"t": {"$date": "2025-10-12T20:37:53.257-05:00"}, "s": "I", "c": "CONTROL", "id": 23403, "svc": "S", "ctx": "initandlisten", "msg": "Build Info", "attr": {"buildInfo": {"version": "3.6.2", "gitVersion": "8dc5cd2a30c4524132e2d44bb314544dc477e611", "modules": [ ], "allocator": "tcmalloc-gperf", "environment": {"distmod": "windows"}}}}
{"t": {"$date": "2025-10-12T20:37:53.257-05:00"}, "s": "I", "c": "CONTROL", "id": 51765, "svc": "S", "ctx": "initandlisten", "msg": "Operating System", "attr": {"os": {"name": "Microsoft Windows 8", "version": "6.2 (build 9200)"}}}
{"t": {"$date": "2025-10-12T20:37:53.257-05:00"}, "s": "I", "c": "CONTROL", "id": 21951, "svc": "S", "ctx": "initandlisten", "msg": "Options set by command line", "attr": {"options": {"net": {"bindIp": "localhost", "port": 27019}, "replication": {"replSet": "configReplSet"}, "sharding": {"clusterRole": "configsvr"}, "storage": {"dbPath": "C:/Users/ASUS/mongodb/config"}}}}
{"t": {"$date": "2025-10-12T20:37:53.265-05:00"}, "s": "I", "c": "STORAGE", "id": 22315, "svc": "S", "ctx": "initandlisten", "msg": "Opening WiredTiger", "attr": {"config": "create, cache_size=7382M, session_max=33000, eviction=(threads_min=4, threads_max=4), config_base=false, statistics=(fast), log=(enabled=true, remove=true, path=journal, compressor=snappy), builtin_extension_config=(zstd=(compression_level=6)), file_manager=(close_idle_time=600, close_scan_interval=10, close_handle_minimum=2000), statistics_log=(wait=0), json_output=(error, message), verbose=[recovery_progress:1, checkpoint_progress:1, compact_progress:1, backup:0, checkpoint:0, compact:0, evict:0, history_store:0, recovery:0, rts:0, salvage:0, tiered:0, timestamp:0, transaction:0, verify:0, log:0], prefetch=(available=true, default=false),"}}
{"t": {"$date": "2025-10-12T20:37:53.290-05:00"}, "s": "I", "c": "WTRECOV", "id": 22430, "svc": "S", "ctx": "initandlisten", "msg": "WiredTiger message", "attr": {"message": {"ts_sec": 1760319473, "ts_usec": 289430, "thread": "20628:140730168411680", "session_name": "txn-recover", "category": "WT_VERB_RECOVERY_PROGRESS", "category_id": 34, "verbose_level": "DEBUG_1", "verbose_level_id": 1, "msg": "recovery log replay has successfully finished and ran for 0 milliseconds"}}}
{"t": {"$date": "2025-10-12T20:37:53.290-05:00"}, "s": "I", "c": "WTRECOV", "id": 22430, "svc": "S", "ctx": "initandlisten", "msg": "WiredTiger message", "attr": {"message": {"ts_sec": 1760319473, "ts_usec": 289430, "thread": "20628:140730168411680", "session_name": "txn-recover", "category": "WT_VERB_RECOVERY_PROGRESS", "category_id": 34, "verbose_level": "DEBUG_1", "verbose_level_id": 1, "msg": "Set global recovery timestamp: (0,"
```

Ilustración 8. Iniciación de de Config Server



Configuración:

- Puerto: 27019
- Rol: Config Server
- Replica Set: configReplSet
- Ruta de datos: C:\Users\ASUS\mongodb\config
- Versión MongoDB: 8.0.13

2.6.6 Inicializar el Config Server (Replica Set)

- Abrir una SEGUNDA ventana de CMD, la primera debe seguir abierta con el Config Server corriendo. En esta nueva ventana, conectarse al Config Server: `mongosh --port 27019`

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

C:\Users\ASUS>mongosh --port 27019
Current Mongosh Log ID: 68ec5ac482764b6275cebea3
Connecting to:  mongodb://127.0.0.1:27019/?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
  2025-10-12T20:37:53.307-05:00: Access control is not enabled for the database. Read and write access to data and conf
  igation is unrestricted
-----

test> |
```

Ilustración 9. Conexión de Config Server

- Ejecutar este comando para inicializar el Replica Set:

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

MongoDB necesita que ese servidor forme parte de un Replica Set, ya que el Config Server almacena toda la información de configuración del clúster shardeado (metadatos sobre qué fragmentos existen, dónde están y cómo se distribuyen los datos).

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

Ilustración 10. Inicialización del Config Server



```
ok: 1,
'$clusterTime': {
  clusterTime: Timestamp({ t: 1760320531, i: 1 }),
  signature: {
    hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA= ', 0),
    keyId: Long('0')
  }
},
operationTime: Timestamp({ t: 1760320531, i: 1 })
}
configReplSet [direct: primary] test> |
```

Ilustración 11. Confirmación del Server Activo

2.6.7 Iniciar los SHARDS (Fragmentos de datos)

- Ahora vamos a levantar los 2 shards.
- Abrir una TERCERA ventana de CMD, Ejecutar este comando para iniciar el Shard 1:
mongod --shardsvr --replSet shard1ReplSet --port 27018 --dbpath "%USERPROFILE%\mongodb\shard1" --bind_ip localhost
- Dejar esta ventana abierta corriendo.



```
Aggregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.image_collection] not found."}, "stats": {}, "cmd": {"agg
regate": "image_collection", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:09.601-05:00"}, "s": "I", "c": "-", "id": 4939300, "svc": "S", "ctx": "monitoring-keys
-for-HMAC", "msg": "Failed to refresh key cache", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.image_collection] not found."}, "stats": {}, "cmd": {"agg
regate": "image_collection", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:10.001-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [local.oplog.rs] not found."}, "stats": {}, "cmd": {"agg
regate": "oplog.rs", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:10.001-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.transactions] not found."}, "stats": {}, "cmd": {"agg
regate": "transactions", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:10.002-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.image_collection] not found."}, "stats": {}, "cmd": {"agg
regate": "image_collection", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:10.802-05:00"}, "s": "I", "c": "-", "id": 4939300, "svc": "S", "ctx": "monitoring-keys
-for-HMAC", "msg": "Failed to refresh key cache", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.image_collection] not found."}, "stats": {}, "cmd": {"agg
regate": "image_collection", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:11.001-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [local.oplog.rs] not found."}, "stats": {}, "cmd": {"agg
regate": "oplog.rs", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:11.001-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.transactions] not found."}, "stats": {}, "cmd": {"agg
regate": "transactions", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
{"t": {"date": "2025-10-12T21:01:11.001-05:00"}, "s": "W", "c": "QUERY", "id": 23799, "svc": "S", "ctx": "ftdc", "msg": "Ag
gregate command executor error", "attr": {"error": {"code": 26, "codeName": "NamespaceNotFound", "errmsg": "Unable to retrieve s
orageStats in $collStats stage :: caused by :: Collection [config.image_collection] not found."}, "stats": {}, "cmd": {"agg
regate": "image_collection", "cursor": {}, "pipeline": [{"collStats": {"storageStats": {"waitForLock": false, "numericOnly": true
}}}], "db": "config"}}}
```

Ilustración 12. Shard 1



- Abrir una CUARTA ventana de CMD, Ejecutar este comando para iniciar el Shard 2:
mongod --shardsvr --replSet shard2ReplSet --port 27017 --dbpath "%USERPROFILE%\mongodb\shard2" --bind_ip localhost
- Dejar esta ventana abierta corriendo.



Ilustración 13.Shard 2

2.6.8 Inicializar los SHARDS (Replica Sets)

- Ahora inicializaremos los 2 shards como Replica Sets. Abrir una QUINTA ventana de CMD y conectarse al Shard 1: **mongosh --port 27018**

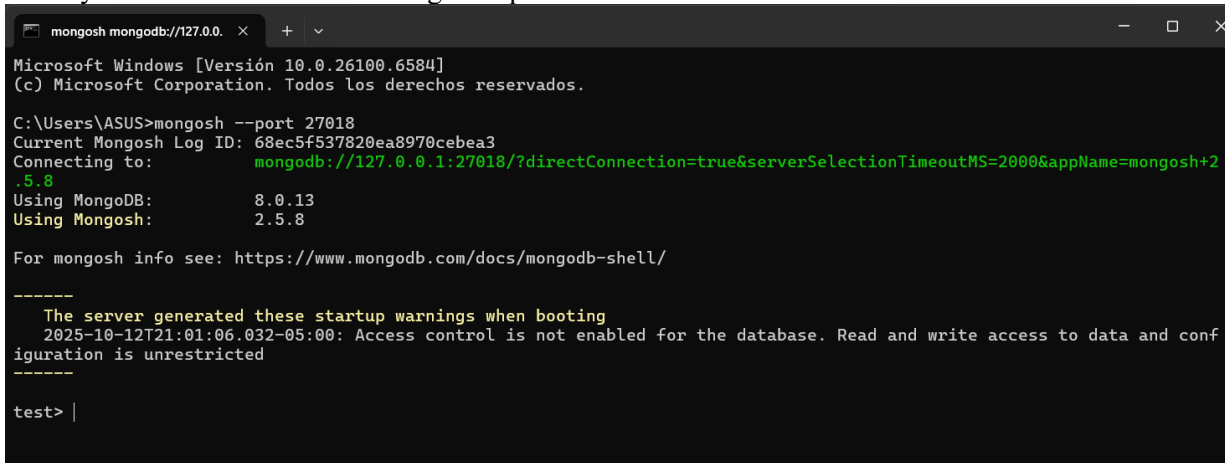


Ilustración 14.Conexión del shard1



- **Inicializa el Shard 1**

```
test> rs.initiate({
...   _id: "shard1ReplSet",
...   members: [{ _id: 0, host: "localhost:27018" }]
... })
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1760321498, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1760321498, i: 1 })
}
shard1ReplSet [direct: primary] test> |
```

Ilustración 15. Inicialización del Replica Set

Este comando crea un Replica Set llamado shard1ReplSet, donde el único miembro inicial es el servidor local que escucha en el puerto 27018.

- **Conectarse al Shard 2: mongosh --port 27017**

```
C:\Users\ASUS>mongosh --port 27017
Current Mongosh Log ID: 68ec60873fca441c44cebea3
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
2025-10-05T22:01:00.848-05:00: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
-----

test> |
```

Ilustración 16. Conexión del shard 2

2.6.9 Iniciar Mongos (El Router)

- Mongos es el enrutador que distribuye las consultas a los shards correctos. Ejecutar en una ventana nueva:

mongos --configdb configReplSet/localhost:27019 --port 27020 --bind_ip localhost.

- Dejar abierta esta ventana

```
Símbolo del sistema - mongo x + v
{"t":{"$date":"2025-10-12T21:27:08.036-05:00"},"s":"I", "c":"NETWORK", "id":23015, "svc":"-", "ctx":"listener","msg":"Listening on","attr":{"address":"127.0.0.1:27020"}}
{"t":{"$date":"2025-10-12T21:27:08.036-05:00"},"s":"I", "c":"NETWORK", "id":23016, "svc":"-", "ctx":"listener","msg":"Waiting for connections","attr":{"port":27020,"ssl":"off"}}
{"t":{"$date":"2025-10-12T21:27:08.036-05:00"},"s":"I", "c":"SHARDING", "id":8423405, "svc":"R", "ctx":"mongosMain","msg":"mongos startup complete","attr":{"Summary of time elapsed":{"Statistics":{"Set up periodic runner":"0 ms","Set up on line certificate status protocol manager":"0 ms","Set up transport layer listener":"0 ms","Initialize global sharding state":"10 ms","Reset the shard registry config connection string":"0 ms","Load global settings from config server":"12 ms","Wait for signing keys":"1001 ms","Pre-cache mongos routing info":"1 ms","Warm up connections to shards":"1 ms","Refresh the balancer configuration":"12 ms","Update read write concern defaults":"3 ms","Start mongos FTDC":"581 ms","Set up script engine":"1 ms","Build user and roles graph":"0 ms","runMongosServer total elapsed time":"1635 ms"}}}}
{"t":{"$date":"2025-10-12T21:27:08.048-05:00"},"s":"I", "c":"SH_REFR", "id":4619902, "svc":"R", "ctx":"CatalogCache-0","msg":"Collection has found to be unsharded after refresh","attr":{"namespace":"config.system.sessions","durationMillis":12}}
{"t":{"$date":"2025-10-12T21:27:08.049-05:00"},"s":"I", "c":"CONTROL", "id":20710, "svc":"R", "ctx":"LogicalSessionC acheRefresh","msg":"Failed to refresh session cache, will try again at the next refresh interval","attr":{"error":"Names paceNotSharded: Expected collection config.system.sessions to be sharded"}}
{"t":{"$date":"2025-10-12T21:27:08.049-05:00"},"s":"I", "c":"CONTROL", "id":20712, "svc":"R", "ctx":"LogicalSessionC acheReap","msg":"Sessions collection is not set up; waiting until next sessions reap interval","attr":{"error":"Namespac eNotSharded: Expected collection config.system.sessions to be sharded"}}
{"t":{"$date":"2025-10-12T21:28:06.425-05:00"},"s":"I", "c":"CONNPOOL", "id":22567, "svc":"-", "ctx":"ShardRegistry", "msg":"Ending idle connection because the pool meets constraints","attr":{"hostAndPort":"localhost:27019","numOpenConns":3}}
{"t":{"$date":"2025-10-12T21:28:06.427-05:00"},"s":"I", "c":"CONNPOOL", "id":22567, "svc":"-", "ctx":"ShardRegistry", "msg":"Ending idle connection because the pool meets constraints","attr":{"hostAndPort":"localhost:27019","numOpenConns":2}}
{"t":{"$date":"2025-10-12T21:28:06.427-05:00"},"s":"I", "c":"CONNPOOL", "id":22567, "svc":"-", "ctx":"ShardRegistry", "msg":"Ending idle connection because the pool meets constraints","attr":{"hostAndPort":"localhost:27019","numOpenConns":1}}
```

Ilustración 17. Distribución de consultas



2.6.10 REPLICACIÓN EN PostgreSQL

Después de realizar la configuración del sharding en MongoDB, se desarrolló la parte del taller enfocada en replicación y control de integridad de datos en PostgreSQL

Creación de la Base de Datos

En pgAdmin se creó una nueva base de datos llamada TRANSPORTE (puede asignarse cualquier nombre).

Esta base contendrá la información de vehículos, conductores, rutas, mantenimientos, gasolina y documentos, todos interrelacionados mediante claves foráneas.

```
CREATE TABLE VEHICULO (  
    ID_VEH SERIAL PRIMARY KEY,  
    MAR_VEH VARCHAR(10) NOT NULL,  
    MOD_VEH VARCHAR(10) NOT NULL,  
    AÑO_VEH INT NOT NULL,  
    ESTADO_DISP CHAR(1));  
  
CREATE TABLE DOCUMENTOS (  
    ID_DOC SERIAL PRIMARY KEY,  
    TIP_DOC VARCHAR(10) NOT NULL,  
    FEC_EXP_DOC DATE NOT NULL,  
    ID_VEH_PER INT NOT NULL,  
    FOREIGN KEY (ID_VEH_PER) REFERENCES VEHICULO(ID_VEH));  
  
CREATE TABLE CONDUCTORES (  
    ID_CONDUCTOR SERIAL PRIMARY KEY,  
    NOM_CON VARCHAR(10) NOT NULL,  
    LIC_CON VARCHAR(10) NOT NULL,  
    FEC_VEN_LIC DATE NOT NULL,  
    ID_VEH_CON INT NOT NULL,  
    FOREIGN KEY (ID_VEH_CON) REFERENCES VEHICULO(ID_VEH));  
  
CREATE TABLE MANTENIMIENTOS (  
    ID_MAN SERIAL PRIMARY KEY,  
    LIC_CON VARCHAR(10) NOT NULL,  
    FEC_VEN_LIC DATE NOT NULL,  
    ID_VEH_PER INT NOT NULL,  
    FOREIGN KEY (ID_VEH_PER) REFERENCES VEHICULO(ID_VEH));  
  
CREATE TABLE GASOLINA (  
    ID_GAS SERIAL PRIMARY KEY,  
    TIP_GAS VARCHAR(10) NOT NULL,  
    DES_GAS VARCHAR(50) NOT NULL,  
    PRECIO_X_GALON DECIMAL(10,2) NOT NULL,  
    ID_VEH_PER INT NOT NULL,  
    FOREIGN KEY (ID_VEH_PER) REFERENCES VEHICULO(ID_VEH));  
  
CREATE TABLE RUTAS (  
    ID_RUT SERIAL PRIMARY KEY,  
    FEC_RUT DATE NOT NULL,  
    ORI_RUT_VIA VARCHAR(10) NOT NULL,  
    DES_RUT_VIA VARCHAR(10) NOT NULL,  
    DIS_RRE_KM DECIMAL(10,2) NOT NULL,  
    DUR_EST_VIA DECIMAL(10,2) NOT NULL,  
    ID_VEH_PER INT NOT NULL,  
    FOREIGN KEY (ID_VEH_PER) REFERENCES VEHICULO(ID_VEH));
```



2.6.10.1 INSERCCION DE DATOS

Se realizaron inserciones de datos de prueba en las tablas para simular la operación del sistema de transporte.

```
Query Query History
1  ✓ INSERT INTO VEHICULO (MAR_VEH, MOD_VEH, AÑO_VEH, ESTADO_DISP) VALUES
2    ('Toyota', 'Corolla', 2020, 'A'),
3    ('Ford', 'Focus', 2018, 'B'),
4    ('Honda', 'Civic', 2019, 'A'),
5    ('Chevrolet', 'Malibu', 2021, 'A'),
6    ('Nissan', 'Sentra', 2022, 'B');
7
8  ✓ INSERT INTO DOCUMENTOS (TIP_DOC, FEC_EXP_DOC, ID_VEH_PER) VALUES
9    ('Seguro', '2023-01-10', 1),
10   ('Permiso', '2023-02-15', 2),
11   ('Licencia', '2023-03-20', 3),
12   ('Inspección', '2023-04-25', 4),
13   ('Registro', '2023-05-30', 5);
14
15  ✓ INSERT INTO CONDUCTORES (NOM_CON, LIC_CON, FEC_VEN_LIC, ID_VEH_CON) VALUES
16    ('Juan', 'ABC123', '2025-01-10', 1),
17    ('Maria', 'DEF456', '2025-02-15', 2),
18    ('Carlos', 'GHI789', '2025-03-20', 3),
19    ('Ana', 'JKL012', '2025-04-25', 4),
20    ('Pedro', 'MNO345', '2025-05-30', 5);
21
22  ✓ INSERT INTO MANTENIMIENTOS (LIC_CON, FEC_VEN_LIC, ID_VEH_PER) VALUES
23    ('ABC123', '2025-01-10', 1),
24    ('DEF456', '2025-02-15', 2),
25    ('GHI789', '2025-03-20', 3),
26    ('JKL012', '2025-04-25', 4),
27    ('MNO345', '2025-05-30', 5);
28
```

Ilustración 18. Inserción de datos

```
INSERT INTO GASOLINA (TIP_GAS, DES_GAS, PRECIO_X_GALON, ID_VEH_PER) VALUES
('Regular', 'Gasolina sin plomo', 3.50, 1),
('Premium', 'Alto octanaje', 4.00, 2),
('Diesel', 'Combustible pesado', 3.80, 3),
('Regular', 'Económico', 3.40, 4),
('Premium', 'Máximo rendimiento', 4.20, 5);

INSERT INTO RUTAS (FEC_RUT, ORI_RUT_VIA, DES_RUT_VIA, DIS_RRE_KM, DUR_EST_VIA, ID_VEH_PER) VALUES
('2023-06-10', 'Ciudad A', 'Ciudad B', 100.5, 2.5, 1),
('2023-07-15', 'Ciudad C', 'Ciudad D', 150.0, 3.0, 2),
('2023-08-20', 'Ciudad E', 'Ciudad F', 200.7, 4.5, 3),
('2023-09-25', 'Ciudad G', 'Ciudad H', 250.3, 5.0, 4),
('2023-10-30', 'Ciudad I', 'Ciudad J', 300.8, 6.5, 5);
```

Ilustración 19. Inserción de datos de tablas faltantes



• **TRIGGER PARA VERIFICAR EL VENCIMIENTO DE LA LICENCIA**

Query Query History

```
1  CREATE OR REPLACE FUNCTION verificar_licencia_vencida()
2  RETURNS TRIGGER AS $$
3  BEGIN
4      -- Si la fecha de vencimiento es menor que la fecha actual, lanzar un error
5      IF NEW.FEC_VEN_LIC < CURRENT_DATE THEN
6          RAISE EXCEPTION 'No se puede registrar un conductor con licencia vencida.';
7      END IF;
8
9      RETURN NEW;
10 END;
11 $$ LANGUAGE plpgsql;
12 CREATE TRIGGER trg_verificar_licencia_vencida
13 BEFORE INSERT OR UPDATE ON CONDUCTORES
14 FOR EACH ROW
15 EXECUTE FUNCTION verificar_licencia_vencida();
16
```

Ilustración 20. Trigger vencimiento de la licencia

TRIGGER PARA VERIFICAR EL CONTROL Y DISPONIBILIDAD DE VEHICULOS

Query Query History

```
1  CREATE OR REPLACE FUNCTION verificar_vehiculo_disponible()
2  RETURNS TRIGGER AS $$
3  DECLARE
4      estado CHAR(1);
5  BEGIN
6      -- Obtener el estado del vehículo
7      SELECT ESTADO_DISP INTO estado FROM VEHICULO WHERE ID_VEH = NEW.ID_VEH_CON;
8
9      -- Si el vehículo no está disponible, generar un error
10 IF estado IS DISTINCT FROM 'A' THEN
11     RAISE EXCEPTION 'El vehículo con ID % no está disponible para asignación.', NEW.ID_VEH_CON;
12 END IF;
13
14 RETURN NEW;
15 END;
16 $$ LANGUAGE plpgsql;
17 CREATE TRIGGER trg_verificar_vehiculo_disponible
18 BEFORE INSERT OR UPDATE ON CONDUCTORES
19 FOR EACH ROW
20 EXECUTE FUNCTION verificar_vehiculo_disponible();
```

Ilustración 21. Trigger verificación de control



TRIGGER PARA VER EL COSTO DEL VIAJE

Query Query History

```
1  CREATE OR REPLACE FUNCTION calcular_costo_viaje()
2  RETURNS TRIGGER AS $$
3  DECLARE
4      precio_litro DECIMAL(10,2);
5  BEGIN
6      -- Obtener el precio por litro de gasolina del vehículo asociado a la ruta
7      SELECT (PRECIO_X_GALON / 3.785) INTO precio_litro
8      FROM GASOLINA g
9      JOIN VEHICULO v ON g.ID_VEH_PER = v.ID_VEH
10     WHERE v.ID_VEH = NEW.ID_VEH_PER
11     LIMIT 1;
12
13     -- Calcular el costo del viaje
14     NEW.COSTO_VIAJE := precio_litro * NEW.DIS_RRE_KM;
15
16     RETURN NEW;
17 END;
18 $$ LANGUAGE plpgsql;
19
20 CREATE TRIGGER trg_calcular_costo_viaje
21 BEFORE INSERT OR UPDATE ON RUTAS
22 FOR EACH ROW
23 EXECUTE FUNCTION calcular_costo_viaje();
24
```

Ilustración 23. Trigger costo de viaje

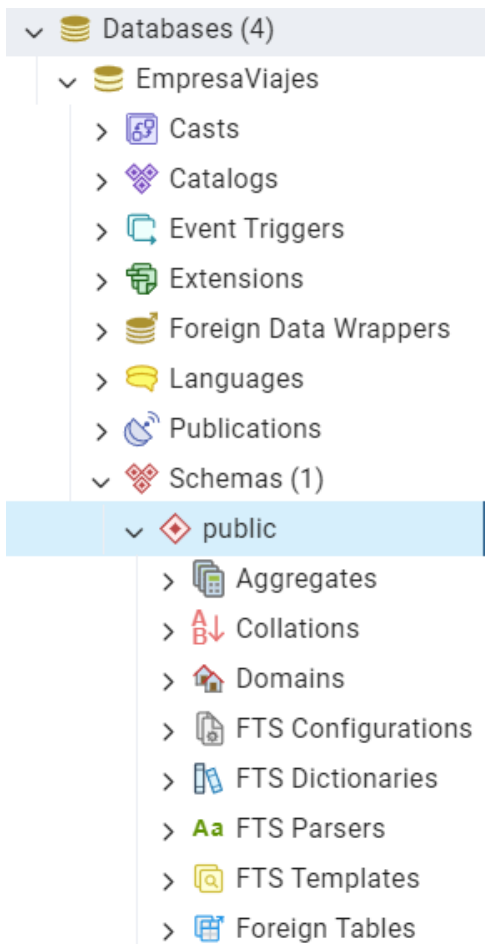


Ilustración 22. BD del proyecto



2.7 Habilidades blandas empleadas en la práctica

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

- Para el desarrollo del taller se utilizó un repositorio en GitHub para el desarrollo colaborativo

<https://github.com/KevinTixilema/Taller-Sharding.git>

```
1  -- =====
2  -- UNIVERSIDAD TÉCNICA DE AMBATO
3  -- SISTEMAS DE BASES DE DATOS DISTRIBUIDOS
4  -- Script 1: Creación de estructura base
5  -- =====
6
7  -- Crear base de datos
8  CREATE DATABASE TRANSPORTE;
9  \c TRANSPORTE;
10
11 -- Tabla VEHICULO
12 CREATE TABLE VEHICULO (
13     ID_VEH SERIAL PRIMARY KEY,
14     MAR_VEH VARCHAR(10) NOT NULL,
15     MOD_VEH VARCHAR(10) NOT NULL,
16     AÑO_VEH INT NOT NULL,
17     ESTADO_DISP CHAR(1)
18 );
19
20 -- Tabla DOCUMENTOS
21 CREATE TABLE DOCUMENTOS (
22     ID_DOC SERIAL PRIMARY KEY,
23     TIP_DOC VARCHAR(10) NOT NULL,
24     FEC_EXP_DOC DATE NOT NULL,
25     ID_VEH_PER INT NOT NULL REFERENCES VEHICULO(ID_VEH)
26 );
27
```

Ilustración 24. Aporte de Cholota Guamán Carlos Sebastián



```
1  -- =====
2  -- Script 2: Tablas adicionales e inserciones
3  -- =====
4
5  -- Tabla MANUTENIMIENTOS
6  CREATE TABLE MANUTENIMIENTOS (
7      ID_MAN SERIAL PRIMARY KEY,
8      LIC_CON VARCHAR(10) NOT NULL,
9      FEC_VEN_LIC DATE NOT NULL,
10     ID_VEH_PER INT NOT NULL REFERENCES VEHICULO(ID_VEH)
11 );
12
13 -- Tabla GASOLINA
14 CREATE TABLE GASOLINA (
15     ID_GAS SERIAL PRIMARY KEY,
16     TIP_GAS VARCHAR(10) NOT NULL,
17     DES_GAS VARCHAR(50) NOT NULL,
18     PRECIO_X_GALON DECIMAL(10,2) NOT NULL,
19     ID_VEH_PER INT NOT NULL REFERENCES VEHICULO(ID_VEH)
20 );
21
22 -- Tabla RUTAS
23 CREATE TABLE RUTAS (
24     ID_RUT SERIAL PRIMARY KEY,
25     FEC_RUT DATE NOT NULL,
26     ORI_RUT_VIA VARCHAR(10) NOT NULL,
27     DES_RUT_VIA VARCHAR(10) NOT NULL,
```

Ilustración 25. Aporte de Mazabanda Pilamunga Diego Abraham

2.8 Conclusiones

- La práctica permitió comprender el funcionamiento del sharding y la replicación en bases de datos distribuidas.
- MongoDB demostró su capacidad para dividir y distribuir datos eficientemente entre varios servidores.
- PostgreSQL permitió reforzar conceptos de integridad y control mediante el uso de triggers.
- La integración de ambas tecnologías mejora la escalabilidad, seguridad y consistencia de los datos.
- Se cumplieron los objetivos del taller, consolidando conocimientos teóricos y prácticos sobre sistemas distribuidos

2.9 Recomendaciones

- Realizar las prácticas en entornos con varios equipos para observar la comunicación real entre nodos.
- Implementar más shards y réplicas para analizar el rendimiento del sistema.
- Utilizar herramientas de monitoreo para verificar el balanceo de carga y la disponibilidad.
- Mantener copias de seguridad frecuentes antes de realizar configuraciones distribuidas.



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE Elige un elemento.
CICLO ACADÉMICO: MARZO – JULIO 2025



- Continuar practicando con ejemplos más complejos para fortalecer el dominio de MongoDB y PostgreSQL.

2.10 Referencias bibliográficas

[1] MongoDB Inc., “Download MongoDB Community Server,” *MongoDB*,
<https://www.mongodb.com/try/download/community> (accessed Oct. 12, 2025).

2.11 Anexos