

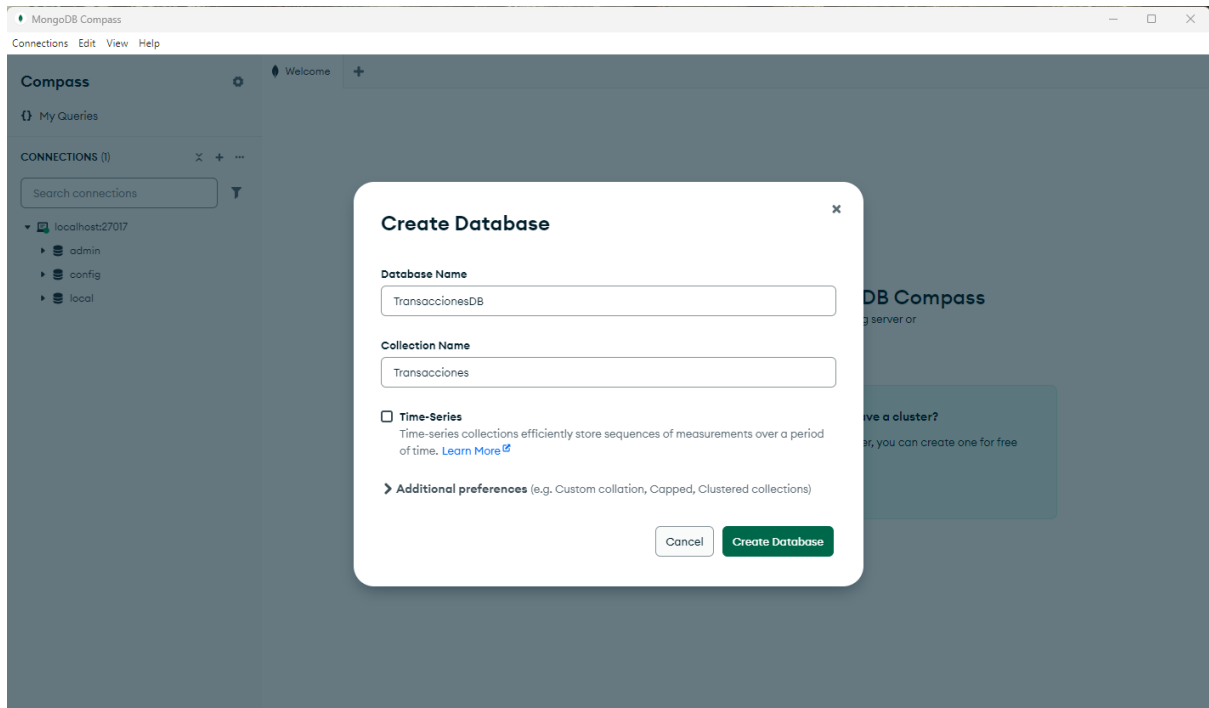
Informa Extra Acerca del Lab2.

<https://www.youtube.com/watch?v=eKXlXSZrJfw> ///instalacion de mongoDB

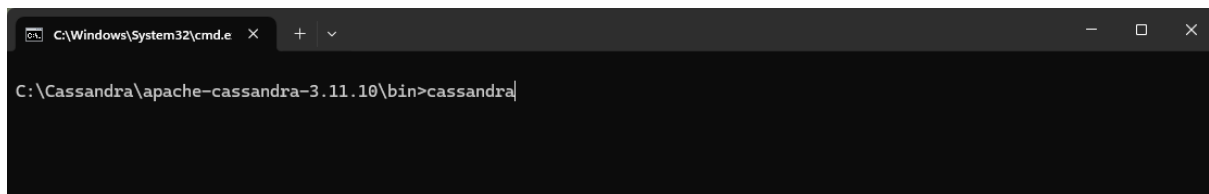
<https://www.youtube.com/watch?v=4tliUDXFxxg&t=143s> ///instalacion cassandra

Crear base de datos en MongoDB

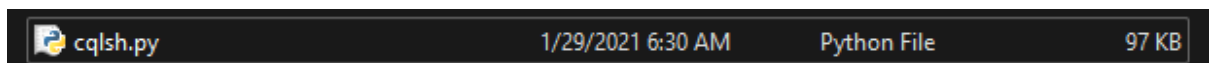
- Nombre base de datos
 - TransaccionesDB
- Nombre de colección
 - Transacciones



Inicio Servidor Cassandra



Se inicia la consola para ejecutar comandos de creación de base de datos y su tabla



```
C:\Python27\python.exe x + v
WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.10 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cqlsh> |
```

Ejecutamos el comando **CREATE KEYSPACE TransaccionesDB WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};** para crear la base de datos con nombre *TransaccionesDB*.

Luego ejecutamos el script **USE TransaccionesDB;** para usar la base de datos *TransaccionesDB*.

```
cqlsh> USE TransaccionesDB;
cqlsh:transaccionesdb>
```

Después utilizamos el comando **CREATE TABLE Transacciones (IdTransaccion TEXT PRIMARY KEY, Usuario TEXT, Fecha TIMESTAMP, Monto DECIMAL, CuentaOrigen TEXT, CuentaDestino TEXT);** para crear la tabla de Transacciones con las columnas de IdTransaccion, Usuario, Fecha, Monto, CuentaOrigen, CuentaDestino

```
cqlsh:transaccionesdb> CREATE TABLE Transacciones (IdTransaccion UUID PRIMARY KEY, Usuario TEXT, Fecha TIMESTAMP, Monto DECIMAL, CuentaOrigen TEXT, CuentaDestino TEXT);
cqlsh:transaccionesdb> |
```

Para asegurarnos que la tabla se creó, utilizamos el comando **DESCRIBE TABLE Transacciones;**

```
cqlsh:transaccionesdb> DESCRIBE TABLE Transacciones;

CREATE TABLE transaccionesdb.transacciones (
  idtransaccion uuid PRIMARY KEY,
  cuentadestino text,
  cuentaorigen text,
  fecha timestamp,
  monto decimal,
  usuario text
) WITH bloom_filter_fp_chance = 0.01
   AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
   AND comment = ''
   AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
   AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
   AND crc_check_chance = 1.0
   AND dclocal_read_repair_chance = 0.1
   AND default_time_to_live = 0
   AND gc_grace_seconds = 864000
   AND max_index_interval = 2048
   AND memtable_flush_period_in_ms = 0
   AND min_index_interval = 128
   AND read_repair_chance = 0.0
   AND speculative_retry = '99PERCENTILE';
```

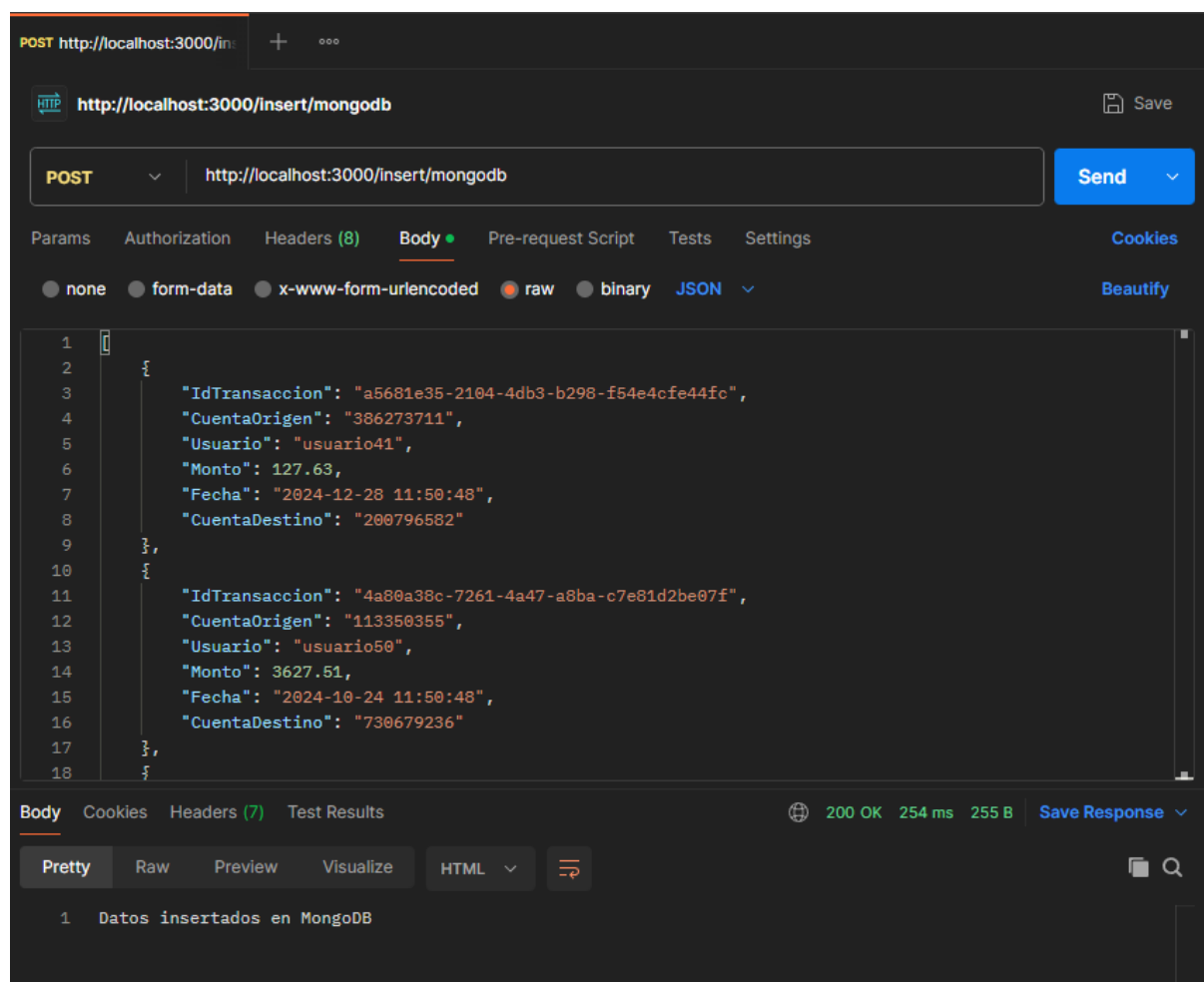
Pruebas para garantizar que las rutas del servidor funcionan correctamente

Tenemos las siguientes rutas en el servidor:

- **Create**
 - <http://localhost:3000/insert/cassandra>
 - <http://localhost:3000/insert/mongodb>
- **Read**
 - <http://localhost:3000/read/cassandra>
 - <http://localhost:3000/read/mongodb>
- **Update**
 - <http://localhost:3000/update/cassandra>
 - <http://localhost:3000/update/mongodb>
- **Delete**
 - <http://localhost:3000/delete/cassandra>
 - <http://localhost:3000/delete/mongodb>

Se designa una ruta para trabajar con cada base de datos para efectuar las pruebas

Prueba Create MongoDB Postman



Prueba Create Cassandra Postman

POST http://localhost:3000/insert/cassandra

Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary JSON

```
1 {
2   {
3     "IdTransaccion": "a5681e35-2104-4db3-b298-f54e4cfe44fc",
4     "CuentaOrigen": "386273711",
5     "Usuario": "usuario41",
6     "Monto": 127.63,
7     "Fecha": "2024-12-28 11:50:48",
8     "CuentaDestino": "200796582"
9   },
10  {
11    "IdTransaccion": "4a80a38c-7261-4a47-a8ba-c7e81d2be07f",
12    "CuentaOrigen": "113350355",
13    "Usuario": "usuario50",
14    "Monto": 3627.51,
15    "Fecha": "2024-10-24 11:50:48",
16    "CuentaDestino": "730679236"
17  },
18 }
```

Body Cookies Headers (7) Test Results 200 OK 874 ms 257 B Save Response

Pretty Raw Preview Visualize HTML

1 Datos insertados en Cassandra

Prueba Read MongoDB Postman

GET http://localhost:3000/read/mongodb

HTTP

http://localhost:3000/read/mongodb

Save

GET

http://localhost:3000/read/mongodb

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (7)

Test Results

200 OK 194 ms 2.03 MB Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   {
3     "_id": "6790116e33f94a171af74153",
4     "IdTransaccion": "a5681e35-2104-4db3-b298-f54e4cfe44fc",
5     "CuentaOrigen": "386273711",
6     "Usuario": "usuario41",
7     "Monto": 127.63,
8     "Fecha": "2024-12-28 11:50:48",
9     "CuentaDestino": "200796582"
10  },
11  {
12    "_id": "6790116e33f94a171af74154",
13    "IdTransaccion": "4a80a38c-7261-4a47-a8ba-c7e81d2be07f",
14    "CuentaOrigen": "113350355",
15    "Usuario": "usuario50",
16    "Monto": 3627.51,
17    "Fecha": "2024-10-24 11:50:48",
18    "CuentaDestino": "730679236"
```

Prueba Read Cassandra Postman

GET http://localhost:3000/read/cassandra

HTTP

http://localhost:3000/read/cassandra

Save

GET

http://localhost:3000/read/cassandra

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (7)

Test Results

200 OK

259 ms

911.53 KB

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   {
3     "idtransaccion": "db7c1dad-8c95-4de8-8f5d-f4ff1c64f33b",
4     "cuentadestino": "316767802",
5     "cuentaorigen": "572033189",
6     "fecha": "2024-11-05T17:50:48.000Z",
7     "monto": "2873.96",
8     "usuario": "usuario63"
9   },
10  {
11    "idtransaccion": "70ce7274-2e69-42ac-8f65-4a268fff1e47",
12    "cuentadestino": "108736681",
13    "cuentaorigen": "173780757",
14    "fecha": "2024-09-12T17:50:48.000Z",
15    "monto": "374.27",
16    "usuario": "usuario82"
17  },
18 }
```

Prueba Update MongoDB Postman

The screenshot shows the Postman interface for a PUT request. The URL is `http://localhost:3000/update/mongodb`. The request method is PUT. The response status is 200 OK, with a response time of 225 ms and a response size of 258 B. The response body is displayed in the 'Body' tab, showing '1 Montos actualizados en MongoDB'.

PUT `http://localhost:3000/update/mongodb` **Send**

Params **Authorization** **Headers (8)** **Body** **Pre-request Script** **Tests** **Settings** **Cookies**

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body **Cookies** **Headers (7)** **Test Results** **200 OK** **225 ms** **258 B** **Save Response**

Pretty **Raw** **Preview** **Visualize** **HTML** **🔍**

1 Montos actualizados en MongoDB

Prueba Update Cassandra Postman

PUT http://localhost:3000/upc

HTTP

http://localhost:3000/update/cassandra

Save

PUT

http://localhost:3000/update/cassandra

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (7)

Test Results

200 OK 336 ms 260 B

Save Response

Pretty

Raw

Preview

Visualize

HTML

1

Montos actualizados en Cassandra

Prueba Delete MongoDB Postman

DEL

http://localhost:3000/delete/

+

...

HTTP

http://localhost:3000/delete/mongodb

Save

DELETE

▼

http://localhost:3000/delete/mongodb

Send

▼

Params

Authorization

Headers (8)

Body ●

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (7)

Test Results

200 OK

109 ms

255 B

Save Response ▼

Pretty

Raw

Preview

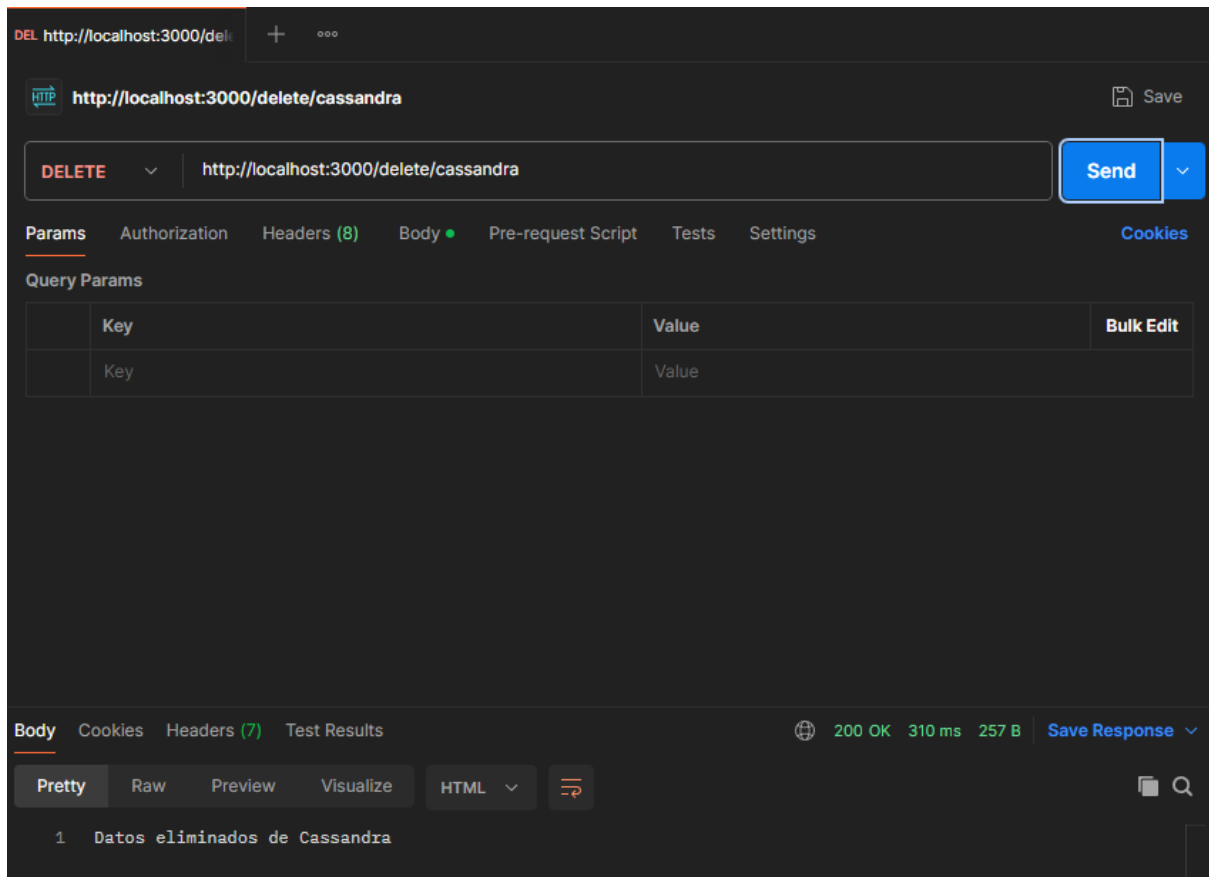
Visualize

HTML ▼

1

Datos eliminados de MongoDB

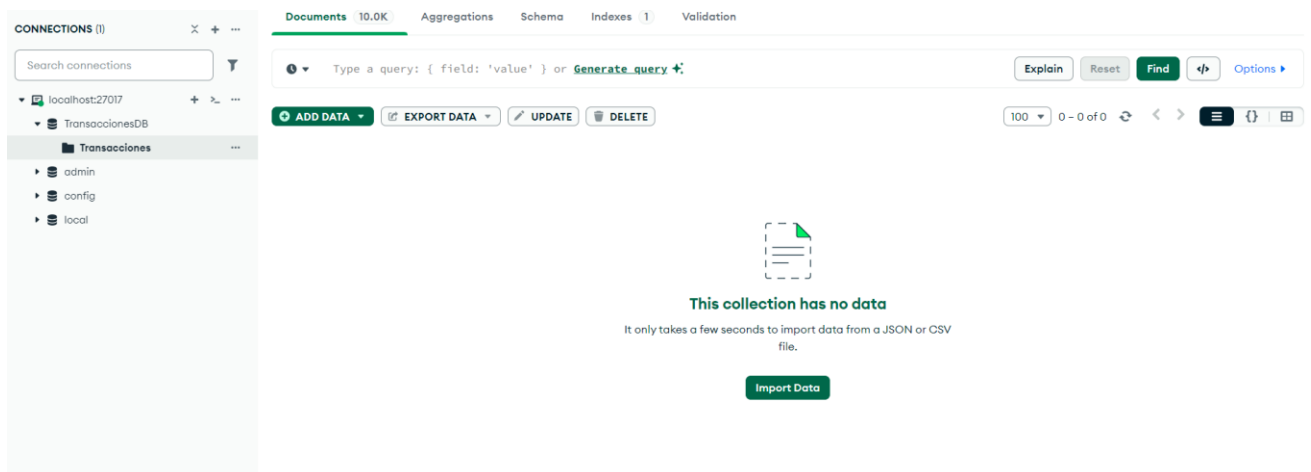
Prueba Delete Cassandra Postman



Pruebas rendimiento utilizando Autocannon

Rendimiento Create

MongoDB antes de insertar datos



CassandraDB antes de insertar datos

```
C:\Python27\python.exe  X + v

WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.10 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cqlsh> use transaccionesdb;
cqlsh:transaccionesdb> select * from transacciones;

 idtransaccion | cuentadestino | cuentaorigen | fecha | monto | usuario
-----+-----+-----+-----+-----+-----
(0 rows)
cqlsh:transaccionesdb> |
```

Imagen del código para ejecutar la prueba de insertar datos para MongoDB y CassandraDB

```
1  const fs = require('fs');
2  const autocannon = require('autocannon');
3
4  //Se lee el archivo json que contiene 10,000 registros para insertar
5  const data = fs.readFileSync('transacciones.json', 'utf8');
6
7  //Configuramos los parámetros de autocannon
8  const instance = autocannon({
9    url: 'http://localhost:3000/insert/mongodb', //URL del servidor
10   connections: 10, //Cantidad de conexiones al mismo tiempo
11   duration: 10, //Segundos que dura la ejecución de las pruebas
12   method: 'POST', //Definimos que es un método POST
13   headers: {
14     'Content-Type': 'application/json' //Indicamos que es de tipo JSON
15   },
16   body: data //Pasamos el JSON leído del archivo transacciones.json para tener datos que insertar
17 });
18
19 //Ejecutamos la prueba de rendimiento
20 autocannon.track(instance);
```

```
1  const fs = require('fs');
2  const autocannon = require('autocannon');
3
4  //Se lee el archivo json que contiene 10,000 registros para insertar
5  const data = fs.readFileSync('transacciones.json', 'utf8');
6
7  //Configuramos los parámetros de autocannon
8  const instance = autocannon({
9    url: 'http://localhost:3000/insert/cassandra', //URL del servidor
10   connections: 10, //Cantidad de conexiones al mismo tiempo
11   duration: 10, //Segundos que dura la ejecución de las pruebas
12   method: 'POST', //Definimos que es un método POST
13   headers: {
14     'Content-Type': 'application/json' //Indicamos que es de tipo JSON
15   },
16   body: data //Pasamos el JSON leído del archivo transacciones.json para tener datos que insertar
17 });
18
19 //Ejecutamos la prueba de rendimiento
20 autocannon.track(instance);
```

Resultado en la base de datos MongoDB

localhost:27017 > TransaccionesDB > Transacciones

Open MongoDB shell

Documents 10.0K Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

100 1 - 100 of 2140000

`_id: ObjectId('6790160e33f94a171af76863')`

`IdTransaccion: "a5681e35-2104-4db3-b298-f54e4cfe44fc"`

`CuentaOrigen: "386273711"`

`Usuario: "usuario41"`

`Monto: 127.63`

`Fecha: "2024-12-28 11:50:48"`

`CuentaDestino: "200796582"`

`_id: ObjectId('6790160e33f94a171af76864')`

`IdTransaccion: "4a80a38c-7261-4a47-a8ba-c7e81d2be07f"`

`CuentaOrigen: "113350355"`

`Usuario: "usuario50"`

`Monto: 3627.51`

`Fecha: "2024-10-24 11:50:48"`

`CuentaDestino: "730679236"`

Resultado en la base de datos Cassandra

```
cqlsh:transaccionesdb> select * from transacciones limit 20;
```

idtransaccion	cuentadestino	cuentaorigen	fecha	monto	usuario
0c6ddb50-f5f9-494c-b53a-6f830da1343b	329470422	673553371	2024-11-27 17:50:48.000000+0000	2546.25	usuario78
5a493c29-519b-4a14-abf8-c0c9c0244145	714197741	639261076	2024-09-20 17:50:48.000000+0000	1695.96	usuario26
2db3f51b-b5eb-4ae8-a6df-045e172f8754	531833978	568885416	2024-10-03 17:50:48.000000+0000	2798.86	usuario39
d5f95a88-ea98-4a52-ab58-709bf73dbae3	200796582	268587601	2024-12-03 17:50:48.000000+0000	3943.61	usuario72
f1012e06-720d-4d24-8579-2b4152373d95	885893168	170910420	2025-01-12 17:50:48.000000+0000	1356.61	usuario11
d342d383-73f6-408a-9a4b-f2dc6d0d60b0	795951760	713698320	2024-04-09 17:50:48.000000+0000	1099.54	usuario72
3ce75f45-9c9f-4662-97c7-b60da44d9474	796300752	329470422	2024-12-24 17:50:48.000000+0000	4112.78	usuario57
be3f6a8c-caf6-4b7d-b9b7-6df214741c35	444430810	142148945	2024-07-29 17:50:48.000000+0000	1406.47	usuario83
f6598428-e1f3-4ac7-8cd6-a427ad968731	200796582	795951760	2024-08-01 17:50:48.000000+0000	4283.36	usuario40
6d27dc88-48b0-4e9c-9253-4c6b1c0eb91d	321895145	459464517	2024-04-23 17:50:48.000000+0000	786.47	usuario29
6522933d-d071-473b-9371-4f3248e23031	250355142	500114343	2024-07-21 17:50:48.000000+0000	1772.33	usuario63
e95d8372-7fff-4e04-85e2-b1d5013948c2	532803336	572033189	2024-06-02 17:50:48.000000+0000	1227.6	usuario8
07b78509-5387-4a0f-8437-1e287fd0b72e	912258195	459464517	2024-09-21 17:50:48.000000+0000	2692.06	usuario9
19e8323a-9fff-468e-aa9a-ed4f171cbd8e	298210817	718543526	2024-12-16 17:50:48.000000+0000	4359.74	usuario84
436b1b95-9f51-4552-ae5-230e825e1b58	107400693	930261100	2025-01-04 17:50:48.000000+0000	547.81	usuario37
71f96f0e-6c36-4565-bd3d-8779d2d3174f	103740891	500114343	2024-04-29 17:50:48.000000+0000	4865.29	usuario54
fc5c31d0-ad31-41cd-988c-ca637f6b9930	302773553	640766197	2024-03-15 17:50:48.000000+0000	3528.76	usuario45
933c604c-4e5c-4904-bab6-b3894500aff5	896708991	386273711	2024-02-16 17:50:48.000000+0000	4970.56	usuario27
0ffbf8c5-6e20-4821-adc0-b60d58207038	362971167	511759974	2024-11-28 17:50:48.000000+0000	3749.99	usuario31
ad3b935c-efde-4a06-82cd-fdd5ab0b72dc	420893859	173780757	2025-01-01 17:50:48.000000+0000	2087.99	usuario34

(20 rows)

Resultado de la prueba de rendimiento para MongoDB

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Insert MongoDB.js"
Running 10s test @ http://localhost:3000/insert/mongodb
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	256 ms	444 ms	770 ms	947 ms	481.32 ms	153.33 ms	1307 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	15	15	20	25	20.4	2.94	15
Bytes/Sec	3.83 kB	3.83 kB	5.1 kB	6.38 kB	5.2 kB	750 B	3.83 kB

Req/Bytes counts sampled once per second.
of samples: 10

214 requests in 10.14s, 52 kB read

Resultado de la prueba de rendimiento para Cassandra

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Insert Cassandra.js"
Running 10s test @ http://localhost:3000/insert/cassandra
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	1223 ms	1581 ms	2524 ms	3351 ms	1686.95 ms	437.32 ms	3351 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	0	0	6	9	5.6	3.17	1
Bytes/Sec	0 B	0 B	1.54 kB	2.31 kB	1.44 kB	815 B	257 B

Req/Bytes counts sampled once per second.
of samples: 10

66 requests in 10.19s, 14.4 kB read

Rendimiento Read

Imagen del código para ejecutar la prueba de leer datos para MongoDB y CassandraDB

```
1  const autocannon = require('autocannon');
2
3  //Se configura los parámetros de autocannon
4  const instance = autocannon({
5      url: 'http://localhost:3000/read/mongodb', //URL del servidor
6      connections: 5, //Cantidad de conexiones simultaneas
7      duration: 10, //Duracion de la prueba
8      method: 'GET', //Se establece el metodo get para obtener datos
9      headers: {
10         'Content-Type': 'application/json' //Se recibe un json
11     }
12 });
13
14 //Se ejecuta la prueba de rendimiento
15 autocannon.track(instance);
```

```
1  const autocannon = require('autocannon');
2
3  //Se configura los parámetros de autocannon
4  const instance = autocannon({
5      url: 'http://localhost:3000/read/cassandra', //URL del servidor
6      connections: 5, //Cantidad de conexiones simultaneas
7      duration: 10, //Duracion de la prueba
8      method: 'GET', //Se establece el metodo get para obtener datos
9      headers: {
10         'Content-Type': 'application/json' //Se recibe un json
11     }
12 });
13
14 //Se ejecuta la prueba de rendimiento
15 autocannon.track(instance);
```

Resultado de la prueba de rendimiento para MongoDB

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Read MongoDB.js"
Running 10s test @ http://localhost:3000/read/mongodb
5 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	322 ms	376 ms	440 ms	466 ms	379.92 ms	32.73 ms	466 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	10	10	14	15	13	1.55	10
Bytes/Sec	21.3 MB	21.3 MB	29.8 MB	31.9 MB	27.6 MB	3.3 MB	21.3 MB

Req/Bytes counts sampled once per second.
of samples: 10

135 requests in 10.11s, 276 MB read

Resultado de la prueba de rendimiento para Cassandra

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Read Cassandra.js"
Running 10s test @ http://localhost:3000/read/cassandra
5 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	806 ms	1946 ms	6371 ms	6371 ms	2933.47 ms	1660.25 ms	6371 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	0	0	0	6	1.5	2.34	4
Bytes/Sec	0 B	0 B	0 B	5.6 MB	1.4 MB	2.18 MB	3.73 MB

Req/Bytes counts sampled once per second.
of samples: 10

20 requests in 10.12s, 14 MB read

Rendimiento Update

Imagen del código para ejecutar la prueba de actualizar datos para MongoDB y CassandraDB

```
1  const autocannon = require('autocannon');
2
3  // se configura los parámetros del autocannon
4  const instance = autocannon({
5    url: 'http://localhost:3000/update/mongodb', //URL del servidor
6    connections: 10, //Cantidad de conexiones simultaneas
7    duration: 10, //Duracion de la prueba
8    method: 'PUT', //Metodo put para actualizar
9    headers: {
10     'Content-Type': 'application/json' //El contenido es tipo json
11   },
12   body: JSON.stringify({
13     Monto: 1997 //Se manda a actualizar el monto a todos los registros
14   })
15 });
16
17 autocannon.track(instance);
```

```
1  const autocannon = require('autocannon');
2
3  // se configura los parámetros del autocannon
4  const instance = autocannon({
5    url: 'http://localhost:3000/update/cassandra', //URL del servidor
6    connections: 10, //Cantidad de conexiones simultaneas
7    duration: 10, //Duracion de la prueba
8    method: 'PUT', //Metodo put para actualizar
9    headers: {
10     'Content-Type': 'application/json' //El contenido es tipo json
11   },
12   body: JSON.stringify({
13     monto: 1997 //Se manda a actualizar el monto a todos los registros
14   })
15 });
16
17 autocannon.track(instance);
```

Resultado en la base de datos MongoDB

➕ ADD DATA

📄 EXPORT DATA

✎ UPDATE

🗑 DELETE

100

1 - 100 of 2140000

↺ ↻ ↷

▶	<pre>_id: ObjectId('6790160e33f94a171af76863') IdTransaccion : "a5681e35-2184-4db3-b298-f54e4cfe44fc" CuentaOrigen : "386273711" Usuario : "usuario41" Monto : 1997 Fecha : "2024-12-28 11:50:48" CuentaDestino : "200796582"</pre>	<div>✎ 🗑 📄 🗑</div>
	<pre>_id: ObjectId('6790160e33f94a171af76864') IdTransaccion : "4a80a38c-7261-4a47-a8ba-c7e81d2be07f" CuentaOrigen : "113350355" Usuario : "usuario50" Monto : 1997 Fecha : "2024-10-24 11:50:48" CuentaDestino : "730679236"</pre>	
	<pre>_id: ObjectId('6790160e33f94a171af76865') IdTransaccion : "120aa09f-56ca-42f9-976a-93e04f7ad5a4" CuentaOrigen : "566384972" Usuario : "usuario82" Monto : 1997 Fecha : "2024-02-11 11:50:48" CuentaDestino : "120967949"</pre>	

```

cqlsh:transaccionesdb> select * from transacciones limit 20;

```

idtransaccion	cuentadestino	cuentaorigen	fecha	monto	usuario
06cddb50-f5f9-494c-b53a-6f830d1a134b	329470422	673553371	2024-11-27 17:50:48.000000+0000	1997	usuario78
5a493c29-519b-da14-abf8-c0c9c0244115	714197741	639261076	2024-09-20 17:50:48.000000+0000	1997	usuario26
2db3f51b-b5eb-dae8-ad6f-045e172f8755	531833978	668885416	2024-10-03 17:50:48.000000+0000	1997	usuario39
d5f95a88-ea98-4a52-ab58-709bf73dbae3	200796582	268587601	2024-12-03 17:50:48.000000+0000	1997	usuario72
f1012e06-720d-4d24-8579-2b4152373d95	885893168	710910420	2025-01-12 17:50:48.000000+0000	1997	usuario11
d342d383-73f6-408a-9a4b-f2dc62d60b0b	795951760	173698320	2024-04-09 17:50:48.000000+0000	1997	usuario72
3ce75f45-9c9f-4662-97c7-b60da449d974	796300752	329470422	2024-12-24 17:50:48.000000+0000	1997	usuario57
be3f6a8c-caf6-4b7d-b9b7-6dfc14741c35	444430810	142148945	2024-07-29 17:50:48.000000+0000	1997	usuario83
f6598428-e1f3-4ac7-8cdc-a427ad9686731	200796582	795951760	2024-08-01 17:50:48.000000+0000	1997	usuario40
6d27dca8-a80b-de9c-9253-dc6b1c0eb91d	321895145	459464517	2024-04-23 17:50:48.000000+0000	1997	usuario29
6522933d-d071-473b-9371-4f32348e23031	2560355142	500114343	2024-07-21 17:50:48.000000+0000	1997	usuario63
e95d8372-7fff-4e04-85e2-b1d5013948c2	532803336	572033189	2024-06-02 17:50:48.000000+0000	1997	usuario8
07b78509-5387-4a0f-8437-1e287fd0b72e	912258195	459464517	2024-09-21 17:50:48.000000+0000	1997	usuario9
19e8323a-9fff-468e-aa9a-ed4f171cbd8e	298210817	718543526	2024-12-16 17:50:48.000000+0000	1997	usuario84
436b1b95-9f51-4552-a1e5-2370e825e1b58	107400693	930261100	2025-01-04 17:50:48.000000+0000	1997	usuario37
71f96f0e-6c36-4565-bd3d-2779ed2d3174f	103740891	500114343	2024-04-29 17:50:48.000000+0000	1997	usuario54
fc5c310d-ad31-41cd-988c-ca637f6b9930	302773553	804766197	2024-03-15 17:50:48.000000+0000	1997	usuario45
933c604c-4e5c-4904-bab6-b3894580aff5	896788991	386273711	2024-02-16 17:50:48.000000+0000	1997	usuario27
0ffbf8c5-6e20-4821-adc0-b60d58207038	362971167	511759974	2024-11-28 17:50:48.000000+0000	1997	usuario31
cad3b935c-efde-4a06-82cd-fdd5ab0b72dc	420893859	173780757	2025-01-01 17:50:48.000000+0000	1997	usuario34

(20 rows)

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Update MongoDB.js"
Running 10s test @ http://localhost:3000/update/mongodb
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	92 ms	111 ms	152 ms	293 ms	114.97 ms	23.03 ms	299 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	70	70	89	92	86.9	6.16	70
Bytes/Sec	18.1 kB	18.1 kB	23 kB	23.7 kB	22.4 kB	1.59 kB	18.1 kB

Req/Bytes counts sampled once per second.
of samples: 10

879 requests in 10.11s, 224 kB read

Resultado de la prueba de rendimiento para Cassandra

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Update Cassandra.js"
Running 10s test @ http://localhost:3000/update/cassandra
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	276 ms	359 ms	658 ms	863 ms	392.33 ms	106.59 ms	896 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	14	14	25	31	25.1	4.16	14
Bytes/Sec	3.64 kB	3.64 kB	6.5 kB	8.06 kB	6.53 kB	1.08 kB	3.64 kB

Req/Bytes counts sampled once per second.
of samples: 10

261 requests in 10.13s, 65.3 kB read

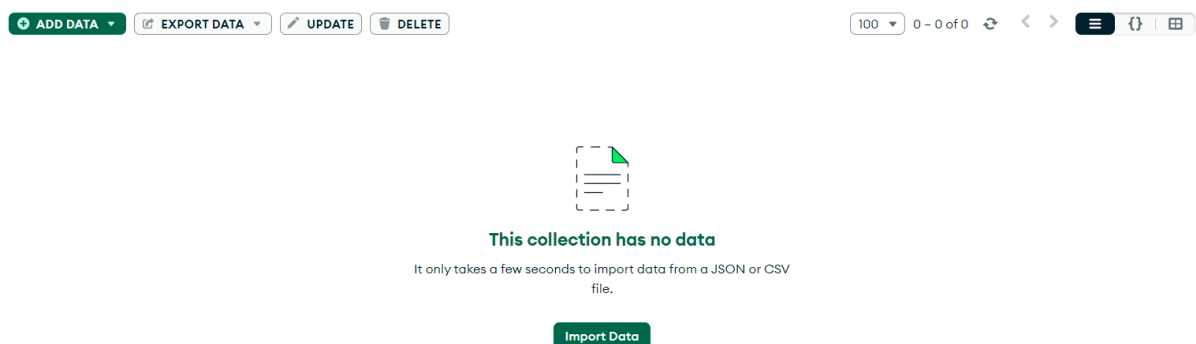
Rendimiento Delete

Imagen del código para ejecutar la prueba de eliminar datos para MongoDB y CassandraDB

```
1  const autocannon = require('autocannon');
2
3  //Configuracion de los parametros
4  const instance = autocannon({
5    url: 'http://localhost:3000/delete/mongodb', //URL
6    connections: 10, //Cantidad de conexiones
7    duration: 10, //Duracion de la prueba
8    method: 'DELETE' //Metodo delete para eliminar
9  });
10
11 autocannon.track(instance);
```

```
1  const autocannon = require('autocannon');
2
3  //Configuracion de los parametros
4  const instance = autocannon({
5    url: 'http://localhost:3000/delete/cassandra', //URL
6    connections: 10, //Cantidad de conexiones
7    duration: 10, //Duracion de la prueba
8    method: 'DELETE' //Metodo delete para eliminar
9  });
10
11 autocannon.track(instance);
```

Resultado en la base de datos MongoDB



Resultado en la base de datos Cassandra

```
cqlsh:transaccionesdb> select * from transacciones limit 20;

 idtransaccion | cuentadestino | cuentaorigen | fecha | monto | usuario
-----+-----+-----+-----+-----+-----
(0 rows)
```

Resultado de la prueba de rendimiento para MongoDB

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Delete MongoDB.js"
Running 10s test @ http://localhost:3000/delete/mongodb
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	3 ms	4 ms	8 ms	10 ms	4.53 ms	3 ms	126 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	1,329	1,329	2,163	2,285	1,997.8	338.56	1,329
Bytes/Sec	339 kB	339 kB	552 kB	583 kB	509 kB	86.4 kB	339 kB

Req/Bytes counts sampled once per second.
of samples: 10

20k requests in 10.03s, 5.09 MB read

Resultado de la prueba de rendimiento para Cassandra

```
C:\Users\lasal\Documents\Base datos avanzadas\Laboratorio 2>node "Pruebas\Prueba Delete Cassandra.js"
Running 10s test @ http://localhost:3000/delete/cassandra
10 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	83 ms	478 ms	1015 ms	1136 ms	479.49 ms	271.67 ms	1210 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	14	14	21	23	20.3	2.58	14
Bytes/Sec	3.6 kB	3.6 kB	5.4 kB	5.91 kB	5.22 kB	660 B	3.6 kB

Req/Bytes counts sampled once per second.
of samples: 10

213 requests in 10.1s, 52.2 kB read

