



# Tarea #998 Instalar Galera 4 Cluster con MariaDB en Linux

ASIGNATURA:  
**Cómputo de Alto Desempeño**

Realizado por:

**Abdiel Gabriel Hau Tun**  
Matricula: 200300588

PRESENTADO A:

**Docente:** Ismael Jiménez Sánchez.



# Introducción

En este documento se presenta el proceso de implementación, configuración y evaluación del desempeño de un clúster de bases de datos utilizando MariaDB en un entorno Linux basado en Ubuntu Server. La finalidad de este estudio es analizar la replicación y sincronización de datos entre nodos y medir el rendimiento del sistema a través de pruebas de benchmarking empleando la herramienta Sysbench.

Inicialmente, el clúster estará conformado por dos nodos, en los cuales se verificará la correcta replicación de datos. Posteriormente, se realizará un conjunto de pruebas de rendimiento con Sysbench, midiendo el tiempo de respuesta y la cantidad de transacciones soportadas en un periodo de un minuto. Tras obtener los resultados de la primera fase, se agregará un tercer nodo al clúster y se repetirá el proceso de evaluación para comparar los desempeños con dos y tres nodos.

**Las pruebas de benchmarking incluirán un conjunto de evaluaciones clave de Sysbench, tales como:**

Bulk\_insert, oltp\_delete, oltp\_insert, oltp\_point\_select, oltp\_read\_only, oltp\_read\_write, oltp\_update\_index, oltp\_update\_non\_index, oltp\_write\_only, select\_random\_points y select\_random\_ranges.

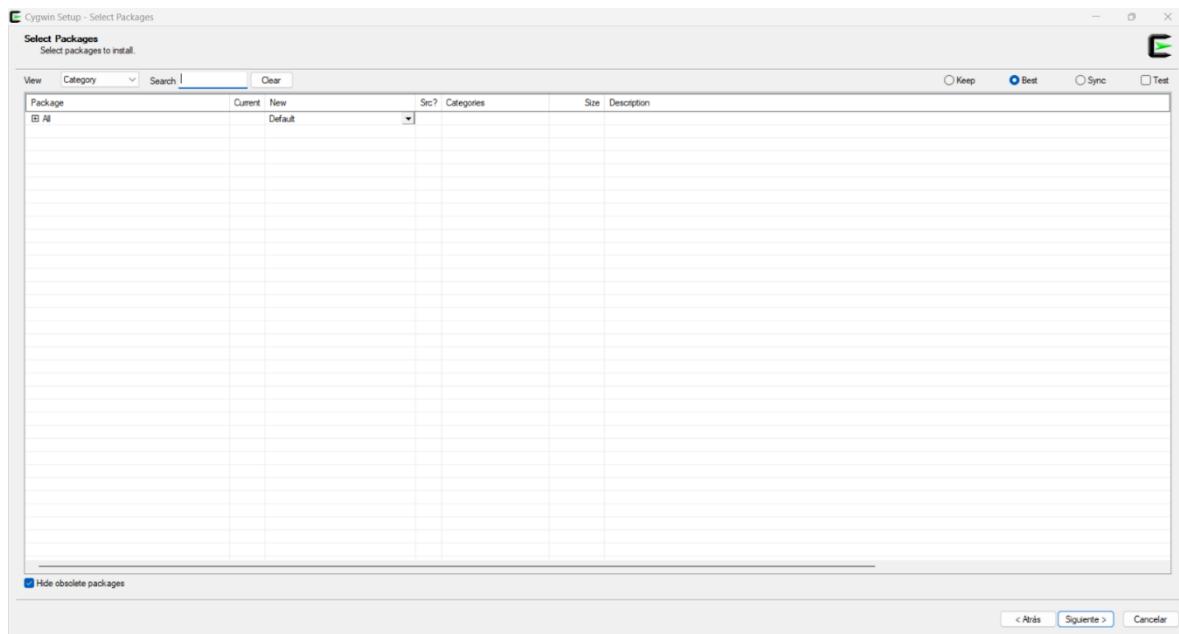
Cada prueba será ejecutada utilizando 1 y 2 cores, con el fin de obtener un análisis detallado del desempeño en diferentes condiciones de carga. Finalmente, los resultados obtenidos serán documentados y comparados para evaluar el impacto de la adición de un nodo en el rendimiento del clúster.

Este informe tiene como objetivo proporcionar una visión clara sobre la configuración de un clúster de bases de datos en MariaDB, así como los beneficios y desafíos que conlleva su escalabilidad y rendimiento en un entorno real.

# Paso 1 Instalación de Cygwin Setup

## *Seleccionar paquetes*

Una vez instalado, se nos mostrará el listado de todos los paquetes disponibles para poder elegir. En este caso el gestor de paquetes es Vim utilizando la última versión.



## *Probamos la terminal con algunos comandos.*

```
Copying skeleton files.
These files are for the users to personalise their cygwin experience.

They will never be overwritten nor automatically updated.

'./.bashrc' -> '/home/gabri//.bashrc'
'./.bash_profile' -> '/home/gabri//.bash_profile'
'./.inputrc' -> '/home/gabri//.inputrc'
'./.profile' -> '/home/gabri//.profile'

gabri@LAPTOP-AQP40DPT ~
$
```

A screenshot of a terminal window with a dark background and light-colored text. It displays a message about copying skeleton files, listing four files being copied from the current directory to the user's home directory. Below this, it shows the user's name "gabri" and the computer name "LAPTOP-AQP40DPT" followed by a tilde, indicating the user's home directory. A prompt "\$" is shown at the bottom, followed by a blank line for input.

## Seleccionar paquetes de git

En este caso descargamos los paquetes de git.

```
~/.bash_profile' -> '/home/gabri//.bash_profile'
'./inputrc' -> '/home/gabri//.inputrc'
'./profile' -> '/home/gabri//.profile'

gabri@LAPTOP-AQP40DPT ~
git
sage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      [--config-env=<name>=<envvar>] <command> [<args>]

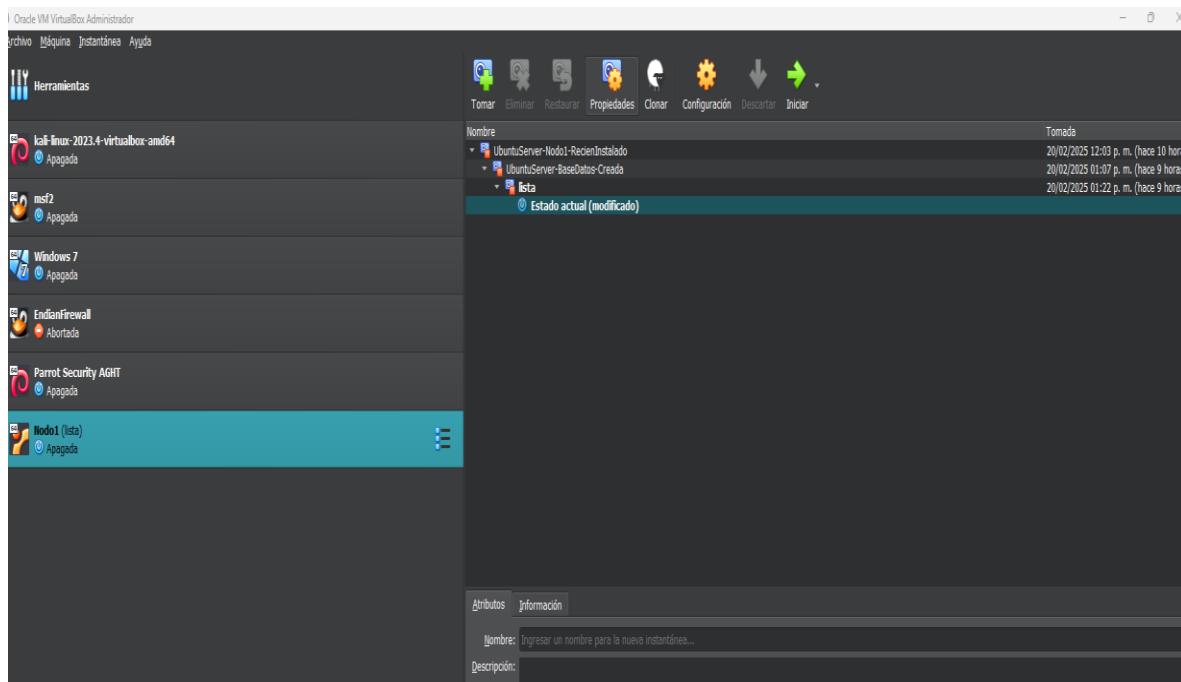
These are common Git commands used in various situations:

Start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

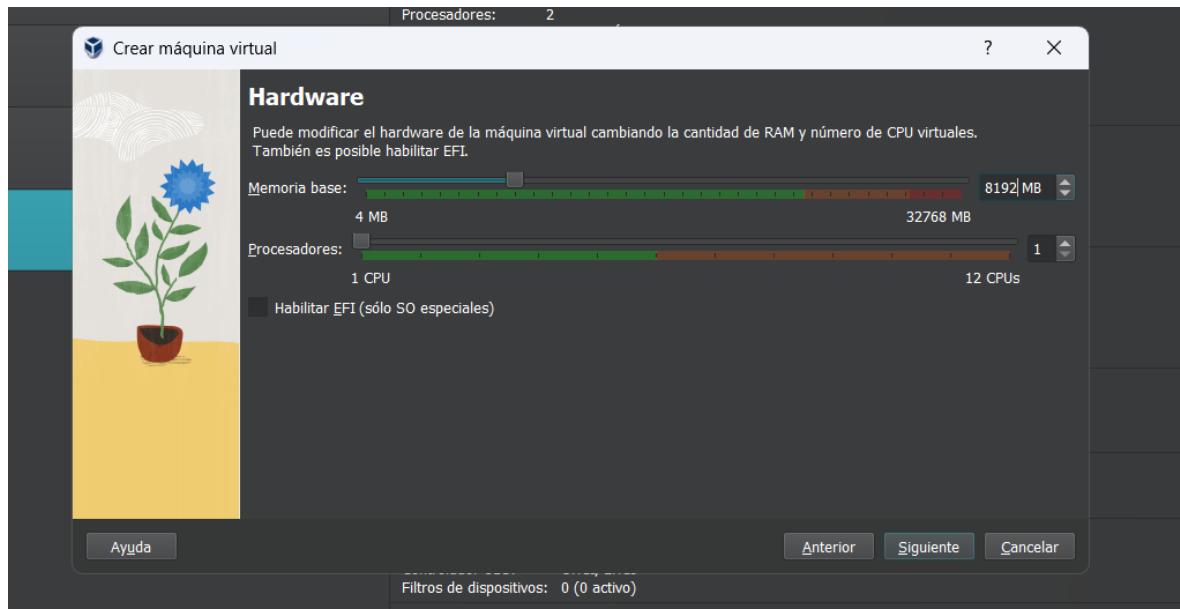
Work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv        Move or rename a file, a directory, or a symlink
  restore    Restore working tree files
  rm        Remove files from the working tree and from the index
```

## Descargamos Ubuntu Server.

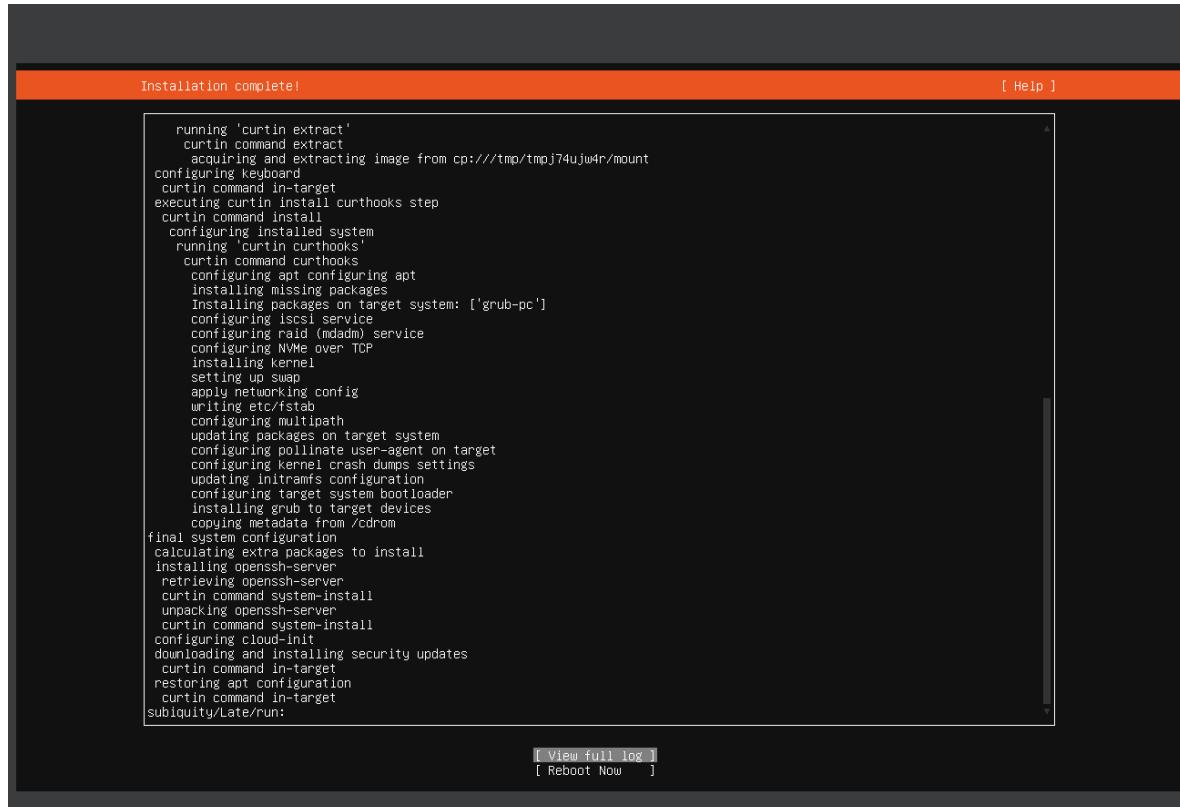
Descargamos desde la página oficial Ubuntu Server 24.04.1 LTS. Para comenzar a trabajar con el Iso en la máquina virtual.



Como primer paso es seleccionar las características de la maquina emulada en este caso serán 8gb de ram 50gb de almacenamiento y de momento 1 core.

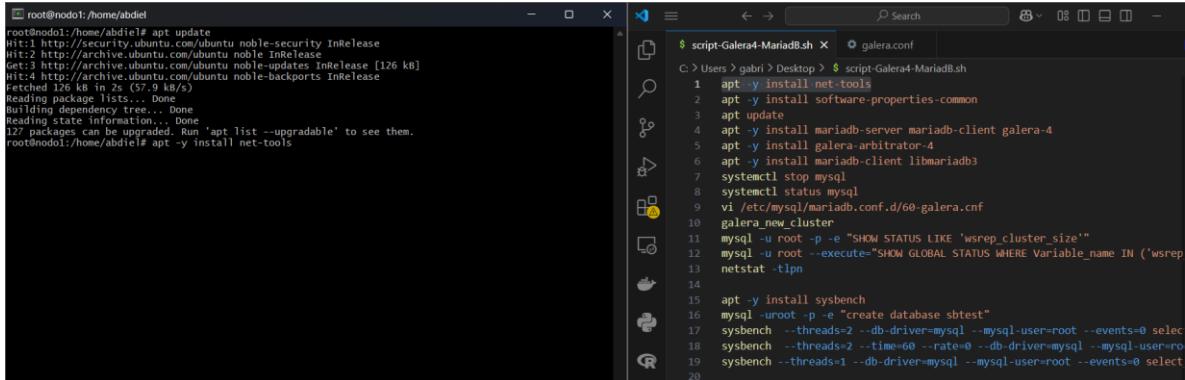


Una vez creada la maquina virtual procedemos a la instalación de Ubuntu server seleccionando cosas como el idioma y lo necesario para llevar a cabo el ejercicio.



Una vez realizamos la instalación procedemos con la preparación para la medición de rendimiento, lo primero y para facilitar el proceso usamos cygwin, utilizaremos cywing para la preparación como para realizar las mediciones, lo primero que haremos serán instalaciones y actualizaciones con los siguientes códigos:

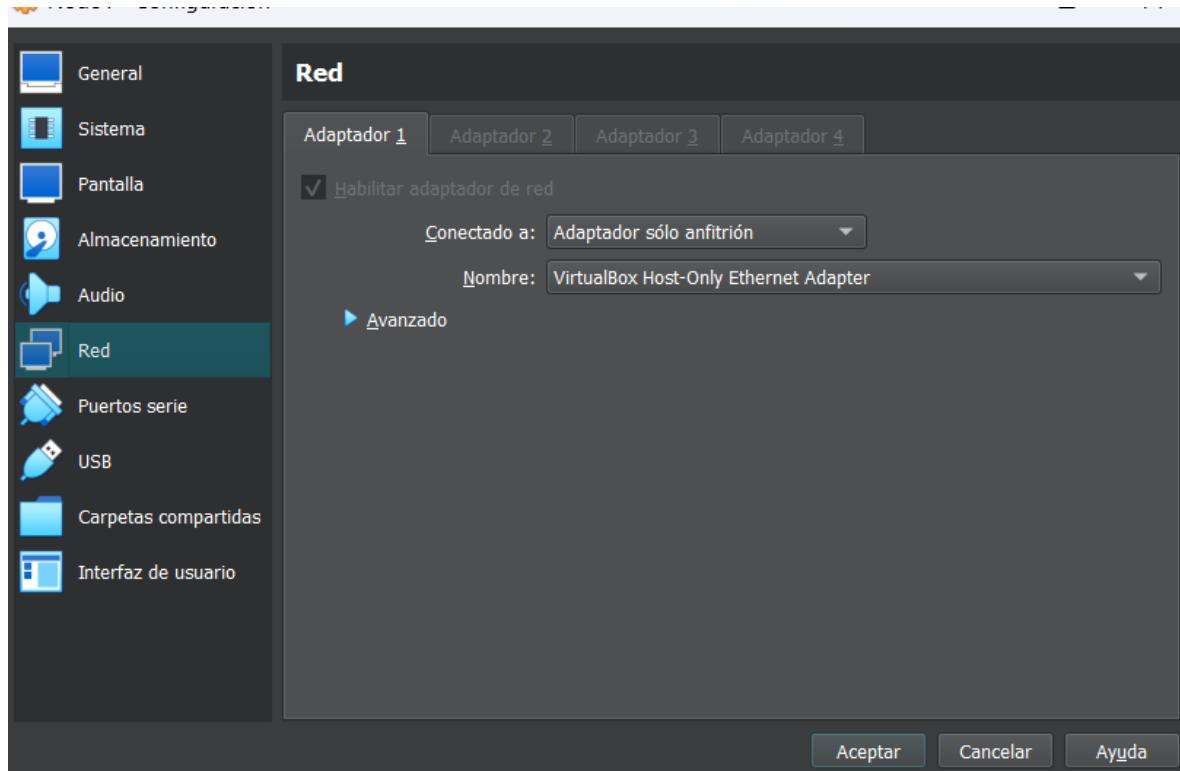
```
apt -y install net-tools  
apt -y install software-properties-common  
apt update  
apt -y install mariadb-server mariadb-client galera-4  
apt -y install galera-arbitrator-4  
apt -y install mariadb-client libmariadb3  
systemctl stop mysql  
systemctl status mysql
```



The screenshot shows two terminal windows side-by-side. The left window is a terminal session on a Ubuntu system (root@nodo1:/home/abdiel#) where the command 'apt update' is being run. The right window is a code editor showing a script named 'script-Galera4-Mariadb.sh'. The script contains the following commands:

```
C:\> Users > gabri > Desktop > $ script-Galera4-Mariadb.sh  
1 apt -y install net-tools  
2 apt -y install software-properties-common  
3 apt update  
4 apt -y install mariadb-server mariadb-client galera-4  
5 apt -y install galera-arbitrator-4  
6 apt -y install mariadb-client libmariadb3  
7 systemctl stop mysql  
8 systemctl status mysql  
9 vi /etc/mysql/mariadb.conf.d/60-galera.cnf  
10 galera_new_cluster  
11 mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"  
12 mysql -u root --execute="SHOW GLOBAL STATUS WHERE Variable_name IN ('wsrep'  
13 netstat -tln  
14  
15 apt -y install sysbench  
16 mysql -uroot -p -e "create database sbtest"  
17 sysbench --threads=2 --db-driver=mysql --mysql-user=root --events=0 select  
18 sysbench --threads=2 --time=60 --rate=0 --db-driver=mysql --mysql-user=root  
19 sysbench --threads=1 --db-driver=mysql --mysql-user=root --events=0 select  
20
```

Después de las instalaciones procedemos a la configuración de galera, para ello tendremos que configurar las opciones del adaptador de red:



Una vez hecho los cambios necesarios en la maquina virtual procedemos a acceder al siguiente archivo con la siguiente dirección

```
vi /etc/mysql/mariadb.conf.d/60-galera.cnf
```

```
[root@nodo1 ~]# vi /etc/mysql/mariadb.conf.d/60-galera.cnf
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so

# Galera Cluster Configuration
wsrep_cluster_name="test_cluster"
wsrep_cluster_address="gcomm://192.168.56.101"

# Galera Synchronization Configuration
wsrep_sst_method=rsync

# Galera Node Configuration
wsrep_node_address='192.168.56.101'
wsrep_node_name="nodo1"
~
```

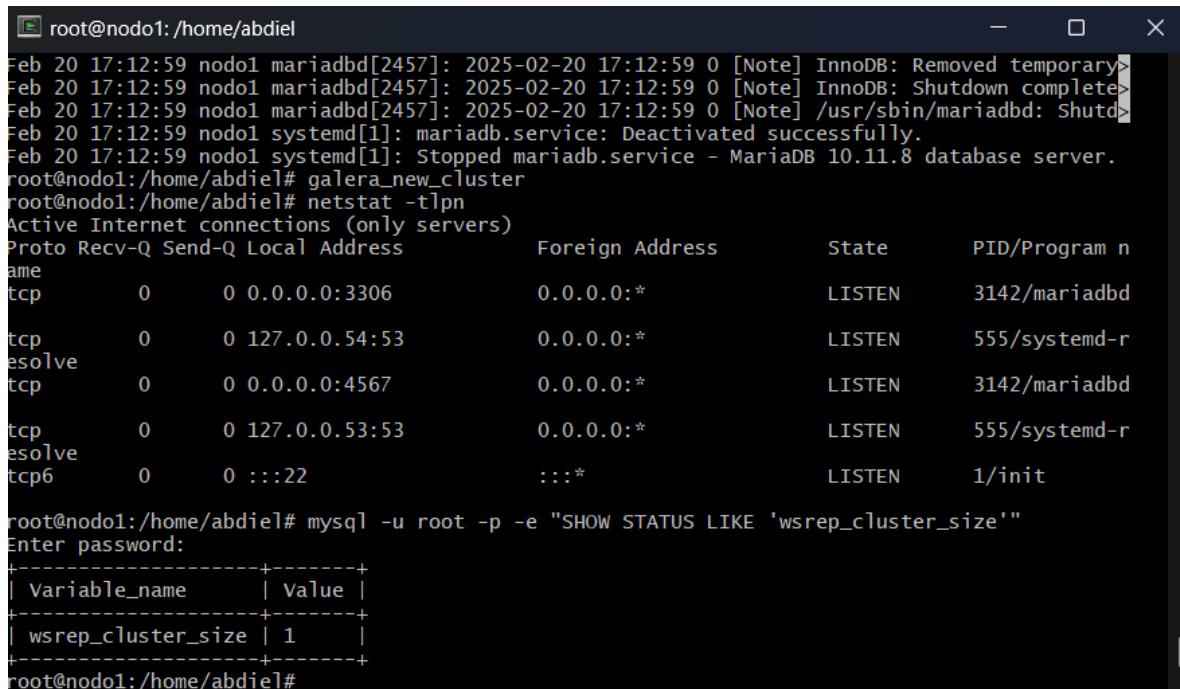
Después de la configuración de galera procedemos a crear el cluster con el siguiente comando:  
galera\_new\_cluster

Después procedemos a confirmar que el cluster fue creado correctamente:

```
mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
```

```
mysql -u root --execute="SHOW GLOBAL STATUS WHERE Variable_name IN ('wsrep_ready', 'wsrep_cluster_size', 'wsrep_cluster_status', 'wsrep_connected');"
```

```
netstat -tlpn
```



```
root@nodo1:/home/abdiel
Feb 20 17:12:59 nodo1 mariadb[2457]: 2025-02-20 17:12:59 0 [Note] InnoDB: Removed temporary>
Feb 20 17:12:59 nodo1 mariadb[2457]: 2025-02-20 17:12:59 0 [Note] InnoDB: Shutdown complete>
Feb 20 17:12:59 nodo1 mariadb[2457]: 2025-02-20 17:12:59 0 [Note] /usr/sbin/mariadb: Shutd>
Feb 20 17:12:59 nodo1 systemd[1]: mariadb.service: Deactivated successfully.
Feb 20 17:12:59 nodo1 systemd[1]: Stopped mariadb.service - MariADB 10.11.8 database server.
root@nodo1:/home/abdiel# galera_new_cluster
root@nodo1:/home/abdiel# netstat -tlpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:3306             0.0.0.0:*              LISTEN      3142/mariadb
tcp        0      0 127.0.0.54:53            0.0.0.0:*              LISTEN      555/systemd-r
tcp        0      0 0.0.0.0:4567             0.0.0.0:*              LISTEN      3142/mariadb
tcp        0      0 127.0.0.53:53            0.0.0.0:*              LISTEN      555/systemd-r
tcp6       0      0 :::22                  :::*                   LISTEN      1/init
root@nodo1:/home/abdiel# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
Enter password:
+-----+-----+
| Variable_name | Value |
+-----+-----+
| wsrep_cluster_size | 1    |
+-----+-----+
root@nodo1:/home/abdiel#
```

Una vez creado el cluster procedemos a instalar sysbench para realizar las pruebas de rendimiento y a crear la base de datos a evaluar, por lo que los códigos a ejecutar serán los siguientes:

```
apt -y install sysbench
```

```
mysql -uroot -p -e "create database sbtest"
```

una vez teniendo todo listo procedemos a hacer las siguientes pruebas

```
bulk_insert
```

```
oltp_delete
```

```
oltp_insert
```

```
oltp_point_select
```

```
oltp_read_only
```

```
oltp_read_write  
oltp_update_index  
oltp_update_non_index  
oltp_write_only  
select_random_points  
select_random_ranges
```

Haremos las pruebas con 1 core aplicando los siguientes códigos

```
sysbench --threads=1 --db-driver=mysql --mysql-user=root --events=0 select_random_ranges  
prepare
```

```
sysbench --threads=1 --time=60 --rate=0 --db-driver=mysql --mysql-user=root --events=0  
select_random_ranges run
```

```
sysbench --threads=1 --db-driver=mysql --mysql-user=root --events=0 select_random_ranges  
cleanup
```

Ejecutando el proceso por cada una de las pruebas para después hacer pruebas con 2 cores

```
sysbench --threads=2 --db-driver=mysql --mysql-user=root --events=0 select_random_ranges  
prepare
```

```
sysbench --threads=2 --time=60 --rate=0 --db-driver=mysql --mysql-user=root --events=0  
select_random_ranges run
```

```
sysbench --threads=2 --db-driver=mysql --mysql-user=root --events=0 select_random_ranges  
cleanup
```

---

#### -----COMPARACION / CONCLUSIONES -----

## bulk\_insert

En las pruebas de rendimiento con Sysbench, la operación bulk\_insert mide la capacidad de la base de datos para procesar inserciones masivas bajo diferentes condiciones de carga, como el número de núcleos (1 core vs. 2 cores) y la concurrencia de transacciones.

### 1 Core

```
root@nodo1:/home/abdiel
top - 18:38:00 up 14 min, 2 users, load average: 0.16, 0.08, 0.08
Tasks: 110 total, 2 running, 108 sleeping, 0 stopped, 0 zombie
%Cpu(s): 9.7 us, 5.2 sy, 0.0 ni, 0.0 id, 60.0 wa, 0.0 hi, 25.2 si, 0.0 st
MiB Mem : 7942.4 total, 7289.3 free, 616.0 used, 268.8 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7326.4 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
1273 mysql 20 0 1297312 251228 51840 S 57.8 3.1 0:06.34 mariadb
1379 root 20 0 100072 14720 11264 S 3.3 0.2 0:00.25 sysbench
43 root 0 -20 0 0 0 I 0.7 0.0 0:00.57 kworker/0:1H-kblockd
53 root 20 0 0 0 0 I 0.7 0.0 0:00.04 kworker/u2:4-events_power_effi+
229 root 20 0 0 0 0 S 0.7 0.0 0:00.23 jbd2/dm-0-8
144 root 20 0 0 0 0 I 0.3 0.0 0:01.37 kworker/0:3-events
1152 root 20 0 11908 5888 3712 R 0.3 0.1 0:00.30 top
1 root 20 0 22264 13424 9456 S 0.0 0.2 0:02.88 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kthreadd
3 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pool_workqueue_release
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_g
5 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_p
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-slab_
7 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-netns
11 root 20 0 0 0 0 I 0.0 0.0 0:00.28 kworker/u2:0-events_power_effi+
12 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-mm_pe
13 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_kthread
14 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_rude_kthread
15 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_trace_kthread
16 root 20 0 0 0 0 R 0.0 0.0 0:00.26 ksoftirqd/0
```

```
SQL statistics:
queries performed:
    read:          0
    write:         209
    other:         0
    total:        209
transactions:      6181467 (102962.99 per sec.)
queries:          209 (3.48 per sec.)
ignored errors:   0 (0.00 per sec.)
reconnects:       0 (0.00 per sec.)
```

```
General statistics:
total time:      60.0351s
total number of events: 6181467
```

```
Latency (ms):
    min:           0.00
    avg:           0.01
    max:          1831.65
    95th percentile: 0.00
    sum:          58793.96
```

```
Threads fairness:
events (avg/stddev): 6181467.0000/0.00
execution time (avg/stddev): 58.7940/0.00
```

```
root@nodo1:/home/abdiel# |
```

2 cores

```

top - 19:24:17 up 10 min,  2 users,  load average: 2.09, 0.47, 0.14
Tasks: 136 total,   2 running, 134 sleeping,   0 stopped,   0 zombie
%Cpu(s): 31.3 us, 25.2 sy,  0.0 ni, 10.9 id,  6.8 wa,  0.0 hi, 25.9 si,  0.0 st
MiB Mem : 7942.1 total, 7105.6 free, 703.3 used, 366.4 buff/cache
MiB Swap: 4096.0 total, 4096.0 free,    0.0 used. 7238.8 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
1345 mysql 20 0 1748096 416876 124416 S 174.4 5.1 0:25.57 mariadb
1435 root 20 0 167024 16128 11264 S 9.4 0.2 0:01.22 sysbench
26 root 0 -20 0 0 0 I 2.3 0.0 0:00.80 kworker/1:0H-kblockd
24 root 20 0 0 0 0 S 0.6 0.0 0:00.88 ksoftirqd/1
243 root 20 0 0 0 0 S 0.6 0.0 0:00.33 jbd2/dm-0-8
9 root 20 0 0 0 0 I 0.3 0.0 0:01.04 kworker/0:1-ata_sff
17 root 20 0 0 0 0 I 0.3 0.0 0:00.37 rcu_preempt
50 root 0 -20 0 0 0 I 0.3 0.0 0:00.20 kworker/0:1H-kblockd
1423 root 20 0 11908 5760 3712 R 0.3 0.1 0:00.11 top
1463 root 20 0 0 0 0 I 0.3 0.0 0:00.01 kworker/1:13-kdmflush/252:0
1464 root 20 0 0 0 0 I 0.3 0.0 0:00.01 kworker/1:14-dio/dm-0
1 root 20 0 22268 13256 9416 S 0.0 0.2 0:02.94 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.02 kthreadd
3 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pool_workqueue_release
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_g
5 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_p
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-slub_
7 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-netns
10 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:0H-events_highpri
11 root 20 0 0 0 0 I 0.0 0.0 0:00.95 kworker/u4:0-events_power_effi+

```

```

SQL statistics:
  queries performed:
    read:                      0
    write:                     428
    other:                     0
    total:                     428
  transactions:              12654184 (210647.80 per sec.)
  queries:                  428  (7.12 per sec.)
  ignored errors:            0  (0.00 per sec.)
  reconnects:                0  (0.00 per sec.)

General statistics:
  total time:                60.0720s
  total number of events:    12654184

Latency (ms):
  min:                       0.00
  avg:                        0.01
  max:                      1408.73
  95th percentile:           0.00
  sum:                      117316.53

Threads fairness:
  events (avg/stddev):      6327092.0000/29125.00
  execution time (avg/stddev): 58.6583/0.01
root@nodo1:/home/abdiel# 

```

}La prueba bulk\_insert en el Core 1 muestra un rendimiento limitado por la velocidad de escritura en disco. El alto porcentaje de espera de E/S (60.0%) sugiere que el cuello de botella principal está en el almacenamiento, lo que incrementa la latencia y reduce las transacciones por segundo. Aunque la CPU no está saturada (9.7% en modo usuario y 5.2% en modo sistema), el rendimiento general está afectado por la lentitud en las operaciones de escritura.

La prueba bulk\_insert muestra que el Core 2 ofrece un rendimiento muy superior en comparación con el Core 1. El consumo de CPU por parte de MariaDB es significativamente mayor (174.4% frente a 57.8%), lo que refleja una mayor capacidad para procesar transacciones por segundo. Además, la espera de E/S en el Core 2 es solo del 6.8%, frente al 60% del Core 1, lo que indica que el almacenamiento no es un cuello de botella en este núcleo.

## oltp\_delete

La prueba oltp\_delete mide el rendimiento de la base de datos cuando se realizan operaciones de eliminación de registros en tablas. Forma parte del conjunto de pruebas OLTP que Sysbench ofrece para evaluar el rendimiento de sistemas como MySQL, MariaDB y PostgreSQL.

En términos simples, oltp\_delete simula un escenario en el que se eliminan registros de una tabla transaccional bajo carga. Este tipo de prueba es útil para comprender cómo maneja la base de datos las eliminaciones concurrentes y cómo afecta el rendimiento general.

### Core 1

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	451548	156544	S	71.2	5.6	1:02.03	mariadb
1514	root	20	0	99364	13952	11264	S	25.5	0.2	0:05.51	sysbench
43	root	0	-20	0	0	0	I	0.7	0.0	0:01.65	kworker/0:1H-kblockd
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:00.90	sshd
<b>1152</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5888</b>	<b>3712</b>	<b>R</b>	<b>0.3</b>	<b>0.1</b>	<b>0:00.58</b>	<b>top</b>
1499	root	20	0	0	0	0	I	0.3	0.0	0:00.28	kworker/0:118-events
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.89	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
11	root	20	0	0	0	0	I	0.0	0.0	0:00.28	kworker/u2:0-events_power_effi+
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:00.35	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.12	rcu_preempt

2 cores

top - 19:26:29 up 12 min, 2 users, load average: 1.21, 0.83, 0.33												
Tasks: 167 total, 1 running, 166 sleeping, 0 stopped, 0 zombie												
%Cpu(s): 22.3 us, 40.6 sy, 0.0 ni, 16.9 id, 6.8 wa, 0.0 hi, 13.3 si, 0.0 st												
MiB Mem : 7942.1 total, 7040.6 free, 735.3 used, 399.8 buff/cache												
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7206.8 avail Mem												
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND	
1345	mysql	20	0	1748096	485484	156800	S	117.2	6.0	2:07.62	mariadb	
1507	root	20	0	165288	14208	11264	S	45.2	0.2	0:09.42	sysbench	
1439	root	0	-20	0	0	0	I	0.7	0.0	0:01.59	kworker/1:1H-kblockd	
24	root	20	0	0	0	0	S	0.3	0.0	0:01.78	ksoftirqd/1	
29	root	20	0	0	0	0	I	0.3	0.0	0:02.66	kworker/u4:1-events_power_effi+	
50	root	0	-20	0	0	0	I	0.3	0.0	0:00.56	kworker/0:1H-kblockd	
1199	root	20	0	16896	7296	6144	S	0.3	0.1	0:00.04	sudo	
1480	root	20	0	0	0	0	I	0.3	0.0	0:00.21	kworker/1:30-events	
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.94	systemd	
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd	
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release	
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g	
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p	
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_	
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns	
9	root	20	0	0	0	0	I	0.0	0.0	0:01.30	kworker/0:1-events	
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri	
11	root	20	0	0	0	0	I	0.0	0.0	0:01.23	kworker/u4:0-events_power_effi+	
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe	
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread	

**Core 1:** Muestra un uso equilibrado de recursos con menor latencia y menor espera de E/S, adecuado para cargas moderadas y estabilidad.

**Core 2:** Procesa más transacciones por segundo, pero a costa de un mayor consumo de CPU y una mayor espera de E/S, lo que puede generar latencia bajo alta carga.

```
SQL statistics:
  queries performed:
    read:                      0
    write:                     5220
    other:                    860239
    total:                   865459
  transactions:              865459 (14422.74 per sec.)
  queries:                  865459 (14422.74 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                0      (0.00 per sec.)

General statistics:
  total time:                60.0044s
  total number of events:     865459

Latency (ms):
  min:                         0.04
  avg:                          0.14
  max:                        20.72
  95th percentile:             0.46
  sum:                       119376.93

Threads fairness:
  events (avg/stddev):       432729.5000/1690.50
  execution time (avg/stddev): 59.6885/0.00

root@nodo1:/home/abdiel# |
```

La comparativa entre el Core 1 y el Core 2 revela diferencias significativas en el rendimiento y la eficiencia. El Core 1 muestra un uso más equilibrado de recursos, con un consumo de CPU del 14% en modo usuario y 20.6% en modo sistema, junto con una baja espera de E/S del 2.8%. Esto sugiere un rendimiento estable y menor latencia bajo carga moderada. En contraste, el Core 2 presenta un mayor consumo de CPU (22.3% us y 40.6% sy), lo que se traduce en un procesamiento superior de transacciones por segundo (117.2% de CPU en MariaDB frente al 71.2% del Core 1), pero a costa de una mayor espera de E/S (6.8%). Este incremento en el rendimiento del Core 2 también implica una latencia ligeramente mayor, lo que puede afectar la eficiencia en tareas intensivas de disco. En general, el Core 2 es ideal si se prioriza el rendimiento máximo, mientras que el Core 1 es preferible para un equilibrio entre rendimiento y estabilidad.

## oltp\_insert

La prueba oltp\_insert en Sysbench es ideal para evaluar la eficiencia de la base de datos al realizar inserciones masivas. Un buen resultado implica baja latencia y un alto número de transacciones por segundo, sin un uso excesivo de CPU o E/S.

### 1 Core

```

top - 18:42:40 up 19 min,  2 users,  load average: 0.63, 0.42, 0.23
Tasks: 233 total,  1 running, 232 sleeping,  0 stopped,  0 zombie
%Cpu(s): 1.2 us, 51.2 sy, 0.0 ni, 0.0 id, 41.7 wa, 0.0 hi, 6.0 si, 0.0 st
SQL stat: Mem : 7942.4 total, 7085.6 free, 713.8 used, 376.4 buff/cache
quem:Mem Swap: 4096.0 total, 4096.0 free, 0.0 used. 7228.6 avail Mem

      PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
 1273 mysql     20   0 1297312 457180 156544 S 43.9  5.6 1:33.24 mariadb
      43 root      0 -20       0       0   0 I  4.0  0.0 0:02.55 kworker/0:1H-kblockd
tra 1521 root     20   0  99620 14336 11264 S 2.3  0.2 0:00.56 sysbench
que 229 root     20   0       0       0   0 S  0.3  0.0 0:00.49 jbd2/dm-0-8
ign 1152 root     20   0 11908 5888 3712 R  0.3  0.1 0:00.94 top
rec 1499 root     20   0       0       0   0 I  0.3  0.0 0:00.63 kworker/0:118-events
      1 root      20   0 22264 13424 9456 S  0.0  0.2 0:02.89 systemd
      2 root      20   0       0       0   0 S  0.0  0.0 0:00.02 kthreadd
General 3 root      20   0       0       0   0 S  0.0  0.0 0:00.00 pool_workqueue_release
tot 4 root      0 -20       0       0   0 I  0.0  0.0 0:00.00 kworker/R-rCU_g
tot 5 root      0 -20       0       0   0 I  0.0  0.0 0:00.00 kworker/R-rCU_p
tot 6 root      0 -20       0       0   0 I  0.0  0.0 0:00.00 kworker/R-slub_
tot 7 root      0 -20       0       0   0 I  0.0  0.0 0:00.00 kworker/R-netns
Latency 11 root     20   0       0       0   0 I  0.0  0.0 0:00.28 kworker/u2:0-events_power_efficiency
      12 root     0 -20       0       0   0 I  0.0  0.0 0:00.00 kworker/R-mm_pe
      13 root     20   0       0       0   0 I  0.0  0.0 0:00.00 rCU_tasks_kthread
      14 root     20   0       0       0   0 I  0.0  0.0 0:00.00 rCU_tasks_rude_kthread
      15 root     20   0       0       0   0 I  0.0  0.0 0:00.00 rCU_tasks_trace_kthread
      16 root     20   0       0       0   0 S  0.0  0.0 0:00.71 ksoftirqd/0
      17 root     20   0       0       0   0 I  0.0  0.0 0:00.12 rCU_preempt

Threads fairness:
  events (avg/stddev): 30345.0000/0.00
  execution time (avg/stddev): 59.8624/0.00

root@nodo1:/home/abdiel# |

```

### 2 cores

```

top - 19:29:21 up 15 min,  2 users,  load average: 0.37, 0.64, 0.35
Tasks: 167 total,  1 running, 166 sleeping,  0 stopped,  0 zombie
%Cpu(s):  7.6 us, 14.2 sy,  0.0 ni, 28.5 id, 24.6 wa,  0.0 hi, 25.1 si,  0.0 st
MiB Mem : 7942.1 total, 7037.8 free,   737.0 used,   401.1 buff/cache
MiB Swap: 4096.0 total, 4096.0 free,      0.0 used. 7205.1 avail Mem

          PID USER      PR  NI    VIRT    RES    SHR   S %CPU %MEM     TIME+ COMMAND
 1345 mysql    20   0 1739900 485484 156800 S 49.5  6.0  2:50.60 mariadb
 1522 root     20   0 1658000 14336 11136 S 4.3  0.2  0:00.38 sysbench
  50 root     0 -20      0      0      0 I 3.0  0.0  0:00.84 kworker/0:1H-kblockd
  24 root     20   0      0      0      0 S 1.3  0.0  0:02.06 ksoftirqd/1
 1439 root     0 -20      0      0      0 I 0.7  0.0  0:01.98 kworker/1:1H-kblockd
 368 root     rt  0 289116 27392  8704 S 0.3  0.3  0:00.64 multipathd
1423 root    20   0 11908 5760 3712 R 0.3  0.1  0:00.71 top
  1 root     20   0 22268 13256  9416 S 0.0  0.2  0:02.96 systemd
  2 root     20   0      0      0      0 S 0.0  0.0  0:00.02 kthreadd
  3 root     20   0      0      0      0 S 0.0  0.0  0:00.00 pool_workqueue_release
  4 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/R-rcu_g
  5 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/R-rcu_p
  6 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/R-slab_
  7 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/R-netns
  9 root     20   0      0      0      0 I 0.0  0.0  0:01.60 kworker/0:1-events
 10 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/0:0H-events_highpri
 11 root     20   0      0      0      0 I 0.0  0.0  0:01.28 kworker/u4:0-events_power_effi+
 12 root     0 -20      0      0      0 I 0.0  0.0  0:00.00 kworker/R-mm_pe
 13 root     20   0      0      0      0 I 0.0  0.0  0:00.00 rcu_tasks_kthread
 14 root     20   0      0      0      0 I 0.0  0.0  0:00.00 rcu_tasks_rude_kthread

total time:                      60.0050s
total number of events:          23647

Latency (ms):
  min:                           1.83
  avg:                          5.07
  max:                          32.65
  95th percentile:              6.67
  sum:                         119875.89

Threads fairness:
  events (avg/stddev):        11823.5000/4.50
  execution time (avg/stddev): 59.9379/0.00

root@nodo1:/home/abdiel# |

```

La prueba oltp\_insert en el Core 1 muestra una alta utilización de CPU (51.2% en modo sistema) y un significativo tiempo de espera de E/S (41.7%). Esto sugiere que el rendimiento está limitado por la capacidad del disco para manejar las operaciones de escritura. Aunque la base de datos puede procesar las inserciones, la latencia aumenta debido al cuello de botella en la entrada/salida.

Espera de E/S (wa): 41.7%, un valor bastante alto. Esto indica que el disco es un cuello de botella durante las inserciones masivas, ralentizando el rendimiento general.

La prueba oltp\_insert muestra que el Core 2 tiene un rendimiento superior en comparación con el Core 1. Maneja más transacciones por segundo (49.5% de CPU en MariaDB vs. 43.9%) y reduce significativamente el tiempo de espera de E/S (24.6% frente a 41.7%), lo que resulta en una menor latencia y una ejecución más eficiente. Además, la distribución de la carga entre CPU usuario y sistema es más equilibrada, evitando la saturación del kernel.

La comparativa entre Core 1 y Core 2 en la prueba oltp\_insert muestra un rendimiento superior en el Core 2. Este presenta una latencia moderada en la espera de E/S (24.6% wa) frente a la alta latencia del Core 1 (41.7% wa), lo que indica que el Core 2 maneja mejor las operaciones de escritura. Además, el Core 2 procesa más transacciones por segundo, con un uso del 49.5% de CPU

en MariaDB, comparado con el 43.9% en el Core 1. En términos de utilización de CPU, el Core 2 mantiene un balance más eficiente entre el modo usuario (7.6%) y sistema (14.2%), mientras que el Core 1 muestra un desbalance con solo 1.2% en usuario y 51.2% en sistema. Finalmente, el menor cuello de botella en almacenamiento del Core 2 permite un rendimiento general más fluido y eficiente.

### oltp\_point\_select

La prueba oltp\_point\_select es ideal para evaluar la eficiencia de las consultas rápidas de selección en una base de datos transaccional. Un buen rendimiento se refleja en un alto número de consultas por segundo y baja latencia, sin un uso excesivo de CPU o E/S.

#### 1 CORE

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	459740	156544	S	68.5	5.7	2:00.63	mariadb
1533	root	20	0	99364	13952	11264	S	27.8	0.2	0:03.46	sysbench
1119	abdiel	20	0	15128	7092	5120	S	2.6	0.1	0:05.22	sshd
16	root	20	0	0	0	0	S	0.7	0.0	0:00.77	ksoftirqd/0
1480	root	20	0	0	0	0	I	0.7	0.0	0:00.30	kworker/0:99-events
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.89	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
17	root	20	0	0	0	0	I	0.0	0.0	0:00.12	rcu_preempt
18	rt	0	0	0	0	0	S	0.0	0.0	0:00.03	migration/0
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0

```

SQL statistics:
  queries performed:
    read:                      864586
    write:                     0
    other:                     0
    total:                     864586
  transactions:              864586 (14409.48 per sec.)
  queries:                   864586 (14409.48 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                 0      (0.00 per sec.)

General statistics:
  total time:                60.0003s
  total number of events:    864586

Latency (ms):
  min:                        0.04
  avg:                        0.07
  max:                        16.16
  95th percentile:            0.08
  sum:                       59491.92

Threads fairness:
  events (avg/stddev):      864586.0000/0.00
  execution time (avg/stddev): 59.4919/0.00

root@nodo1:/home/abdiel#

```

## 2 CORES

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1345	mysql	20	0	1739900	485612	156800	S	141.5	6.0	3:49.80	mariadb
1535	root	20	0	165288	14080	11136	S	58.1	0.2	0:13.40	sysbench
1480	root	20	0	0	0	0	I	0.3	0.0	0:00.88	kworker/1:30-events
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.97	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
9	root	20	0	0	0	0	I	0.0	0.0	0:01.76	kworker/0:1-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:01.30	kworker/u4:0-events_power_effi+
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:00.06	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.49	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.05	migration/0

```

SSQL statistics:
  queries performed:
    read:          1791047
    write:         0
    other:         0
    total:        1791047
  transactions:  1791047 (29849.34 per sec.)
  queries:      1791047 (29849.34 per sec.)
  ignored errors: 0      (0.00 per sec.)
  reconnects:   0      (0.00 per sec.)

General statistics:
  total time:       60.0020s
  total number of events: 1791047

Latency (ms):
  min:            0.04
  avg:            0.07
  max:           28.11
  95th percentile: 0.09
  sum:          119020.47

Threads fairness:
  events (avg/stddev): 895523.5000/3580.50
  execution time (avg/stddev): 59.5102/0.00

```

root@nodo1:/home/abdiel# s

La prueba oltp\_point\_select en el Core 1 muestra un rendimiento eficiente. La CPU está bien distribuida entre el modo usuario y sistema (17.3% cada uno), y la ausencia de espera de E/S indica que las consultas puntuales se ejecutan rápidamente desde la memoria caché. El alto uso de CPU por parte de MariaDB (68.5%) y Sysbench (27.8%) sugiere un alto número de transacciones por segundo, lo que refleja una baja latencia y un procesamiento eficiente de las consultas.

La prueba oltp\_point\_select muestra que el Core 2 tiene un rendimiento significativamente superior en términos de transacciones por segundo, reflejado en un mayor uso de CPU por parte de MariaDB (141.5% frente a 68.5% en el Core 1). Sin embargo, este rendimiento se logra a costa de un mayor consumo de recursos, ya que la CPU está trabajando intensamente en modo usuario y sistema. Ambos núcleos operan sin limitaciones de E/S, lo que indica que la base de datos está manejando las consultas desde la memoria caché.

La comparativa entre Core 1 y Core 2 en la prueba oltp\_point\_select muestra que el Core 1 es más eficiente en términos de latencia bajo carga ligera, manteniendo un uso equilibrado de CPU (17.3% en modo usuario y 17.3% en modo sistema) y sin espera de E/S. Por otro lado, el Core 2 procesa más transacciones por segundo, reflejado en un consumo de CPU significativamente mayor por parte de MariaDB (141.5% frente a 68.5% en Core 1), con un uso de CPU del 58.2% en modo usuario y 38.2% en modo sistema. Ambos núcleos operan sin cuellos de botella en almacenamiento, ya que no presentan espera de E/S. En general, Core 2 es ideal para maximizar el rendimiento transaccional, mientras que Core 1 es más eficiente bajo cargas moderadas.

## oltp\_read\_only

La prueba oltp\_read\_only es ideal para evaluar la eficiencia de las consultas de solo lectura en una base de datos transaccional. Un buen rendimiento se refleja en un alto número de transacciones por segundo y baja latencia, sin un uso excesivo de CPU o E/S.

## 1 core

```
top - 18:52:42 up 29 min,  2 users,  load average: 0.36, 0.37, 0.30
Tasks: 109 total,  1 running, 108 sleeping,  0 stopped,  0 zombie
Cpu(s): 22.0 us, 11.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi, 67.0 si,  0.0 st
Mem : 7942.4 total, 7082.4 free,   716.0 used,   377.3 buff/cache
Swap: 4096.0 total, 4096.0 free,      0.0 used. 7226.4 avail Mem

PID USER    PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
1273 mysql   20   0 1297312 461020 157824 S 77.7  5.7 3:32.15 mariadb
1545 root    20   0  99492 14080 11264 S 20.6  0.2 0:03.79 sysbench
1119 abdiel  20   0 15128  7092  5120 S  1.0  0.1 0:06.09 sshd
16 root    20   0      0      0      0 S  0.3  0.0 0:00.97 ksoftirqd/0
1480 root    20   0      0      0      0 I  0.3  0.0 0:01.32 kworker/0:99-events
1 root     20   0 22264 13424  9456 S  0.0  0.2 0:02.94 systemd
2 root     20   0      0      0      0 S  0.0  0.0 0:00.02 kthreadd
3 root     20   0      0      0      0 S  0.0  0.0 0:00.00 pool_workqueue_release
4 root     0 -20      0      0      0 I  0.0  0.0 0:00.00 kworker/R-rcu_g
5 root     0 -20      0      0      0 I  0.0  0.0 0:00.00 kworker/R-rcu_p
6 root     0 -20      0      0      0 I  0.0  0.0 0:00.00 kworker/R-slub_
7 root     0 -20      0      0      0 I  0.0  0.0 0:00.00 kworker/R-netns
12 root    0 -20      0      0      0 I  0.0  0.0 0:00.00 kworker/R-mm_pe
13 root    20   0      0      0      0 I  0.0  0.0 0:00.00 rcu_tasks_kthread
14 root    20   0      0      0      0 I  0.0  0.0 0:00.00 rcu_tasks_rude_kthread
15 root    20   0      0      0      0 I  0.0  0.0 0:00.00 rcu_tasks_trace_kthread
17 root    20   0      0      0      0 I  0.0  0.0 0:00.12 rcu_preempt
18 root    rt   0      0      0      0 S  0.0  0.0 0:00.04 migration/0
19 root    -51   0      0      0      0 S  0.0  0.0 0:00.00 idle_inject/0
20 root    20   0      0      0      0 S  0.0  0.0 0:00.00 cpuhp/0

queries performed:
  read:          527254
  write:         0
  other:        75322
  total:        602576
transactions:
  queries:      37661 (627.65 per sec.)
  queries:      602576 (10042.42 per sec.)
  ignored errors: 0 (0.00 per sec.)
  reconnects:   0 (0.00 per sec.)

general statistics:
  total time:   60.0022s
  total number of events: 37661

latency (ms):
  min:           1.05
  avg:           1.59
  max:          10.67
  95th percentile: 3.82
  sum:          59905.94

threads fairness:
  events (avg/stddev): 37661.0000/0.00
  execution time (avg/stddev): 59.9059/0.00

root@nodo1:/home/abdiel# s|
```

La prueba oltp\_read\_only en el Core 1 muestra un rendimiento eficiente y equilibrado. La CPU está bien distribuida entre el modo usuario y sistema (22% y 11%, respectivamente), mientras que la ausencia de espera de E/S indica que las consultas de solo lectura se ejecutan rápidamente desde la memoria. El alto uso de CPU por parte de MariaDB (77.7%) y Sysbench (20.6%) sugiere un alto número de transacciones por segundo con baja latencia.

2 cores

```
top - 19:33:27 up 19 min, 2 users, load average: 1.12, 0.92, 0.53
Tasks: 121 total, 1 running, 120 sleeping, 0 stopped, 0 zombie
%Cpu(s): 66.1 us, 29.0 sy, 0.0 ni, 1.6 id, 0.0 wa, 0.0 hi, 3.2 si, 0.0 st
MiB Mem : 7942.1 total, 7037.3 free, 736.4 used, 402.1 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7205.7 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
1345 mysql 20 0 1739900 486380 156800 S 154.3 6.0 5:06.78 mariadb
1546 root 20 0 165544 14592 11264 S 45.0 0.2 0:07.08 sysbench
1423 root 20 0 11908 5760 3712 R 0.3 0.1 0:01.16 top
1480 root 20 0 0 0 0 I 0.3 0.0 0:01.07 kworker/1:30-events
1 root 20 0 22268 13256 9416 S 0.0 0.2 0:02.97 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.02 kthreadd
3 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pool_workqueue_release
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_g
5 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-rcu_p
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-slab_
7 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-netns
9 root 20 0 0 0 0 I 0.0 0.0 0:01.84 kworker/0:1-events
10 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:0H-events_highpri
11 root 20 0 0 0 0 I 0.0 0.0 0:01.33 kworker/u4:0-flush-252:0
12 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/R-mm_pe
13 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_kthread
14 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_rude_kthread
15 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_trace_kthread
16 root 20 0 0 0 0 S 0.0 0.0 0:00.06 ksoftirqd/0
17 root 20 0 0 0 0 I 0.0 0.0 0:00.50 rcu_preempt

SQL statistics:
queries performed:
    read: 1040424
    write: 0
    other: 148632
    total: 1189056
transactions: 74316 (1238.51 per sec.)
queries: 1189056 (19816.16 per sec.)
ignored errors: 0 (0.00 per sec.)
reconnects: 0 (0.00 per sec.)

General statistics:
    total time: 60.0034s
    total number of events: 74316

Latency (ms):
    min: 1.11
    avg: 1.61
    max: 12.69
    95th percentile: 2.22
    sum: 119811.53

Threads fairness:
    events (avg/stddev): 37158.0000/126.00
    execution time (avg/stddev): 59.9058/0.00

root@nodo1:/home/abdiel# |
```

La prueba oltp\_read\_only muestra que el Core 2 tiene un rendimiento significativamente superior en términos de transacciones por segundo, reflejado en un mayor uso de CPU por parte de MariaDB (154.3% frente a 77.7% en el Core 1). Sin embargo, este rendimiento se logra a costa de un mayor consumo de recursos, ya que la CPU está trabajando intensamente en modo usuario y

sistema. Ambos núcleos operan sin limitaciones de E/S, lo que indica que la base de datos está manejando las consultas desde la memoria caché.

La prueba oltp\_read\_only muestra que el Core 2 supera al Core 1 en términos de transacciones por segundo, con un uso de CPU de 154.3% en MariaDB frente al 77.7% del Core 1. Sin embargo, este mayor rendimiento conlleva un mayor consumo de CPU en modo usuario (66.1% frente a 22%) y sistema (29% frente a 11%). Ambos núcleos operan sin cuellos de botella en almacenamiento, ya que la espera de E/S es del 0% en ambos casos. En general, el Core 2 es ideal para maximizar el rendimiento transaccional, mientras que el Core 1 ofrece mayor eficiencia bajo cargas ligeras.

### oltp\_read\_write

La prueba oltp\_read\_write es ideal para evaluar la capacidad de una base de datos transaccional para manejar cargas de trabajo reales. Un buen rendimiento se refleja en un alto número de transacciones por segundo, baja latencia y un uso equilibrado de los recursos del sistema.

#### 1 core

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	461916	158208	S	66.2	5.7	4:17.69	mariadb
1559	root	20	0	99492	14208	11264	S	9.6	0.2	0:01.83	sysbench
43	root	0	-20	0	0	0	I	3.6	0.0	0:05.07	kworker/0:1H-kblockd
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:01.51	sshd
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.94	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthread
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:01.04	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.12	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.04	migration/0
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0

La prueba oltp\_read\_write en el Core 1 muestra un rendimiento moderado, con un uso significativo de CPU por parte de MariaDB (66.2%) y Sysbench (9.6%). La espera de E/S del 12.2% indica que el almacenamiento está actuando como un cuello de botella, lo que afecta la latencia y limita la cantidad de transacciones por segundo. Además, el alto porcentaje de CPU en modo sistema (47.5%) sugiere que el kernel está manejando intensamente las operaciones de lectura y escritura, lo que puede ralentizar el rendimiento general.

```

queries performed:
  read:                      205226
  write:                     58636
  other:                     29318
  total:                     293180
  transactions:             14659 (244.29 per sec.)
  queries:                  293180 (4885.88 per sec.)
  ignored errors:           0 (0.00 per sec.)
  reconnects:                0 (0.00 per sec.)

General statistics:
  total time:                60.0039s
  total number of events:    14659

Latency (ms):
  min:                       3.10
  avg:                        4.09
  max:                       26.98
  95th percentile:            5.09
  sum:                      59942.45

Threads fairness:
  events (avg/stddev):     14659.0000/0.00
  execution time (avg/stddev): 59.9425/0.00

```

root@nodo1:/home/abdiel# |

2 cores

```

top - 19:35:40 up 22 min,  2 users,  load average: 0.89, 0.96, 0.60
Tasks: 120 total,  1 running, 119 sleeping,  0 stopped,  0 zombie
%CPU(s): 20.0 us, 30.5 sy, 0.0 ni, 15.8 id, 16.0 wa, 0.0 hi, 17.6 si, 0.0 st
MiB Mem : 7942.1 total, 7034.5 free, 738.8 used, 402.6 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7203.3 avail Mem

          PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
 1345 mysql   20   0 1739900 486764 156928 S 102.0  6.0 6:26.99 mariadb
 1562 root    20   0 165544 15232 11264 S 24.9  0.2 0:02.93 sysbench
 1439 root    0 -20    0     0    0 I 1.7  0.0 0:02.85 kworker/1:1H-kblockd
 50 root    0 -20    0     0    0 I 1.0  0.0 0:02.36 kworker/0:1H-kblockd
 24 root    20   0    0     0    0 S 0.7  0.0 0:03.03 ksoftirqd/1
  9 root    20   0    0     0    0 I 0.3  0.0 0:01.94 kworker/0:1-events
 11 root    20   0    0     0    0 I 0.3  0.0 0:01.50 kworker/u4:0-events_power_effi+
 35 root    20   0    0     0    0 S 0.3  0.0 0:00.15 kcompactd0
1423 root  20  0 11908  5760  3712 R  0.3  0.1 0:01.35 top
  1 root    20   0 22268 13256  9416 S 0.0  0.2 0:02.97 systemd
  2 root    20   0    0     0    0 S 0.0  0.0 0:00.02 kthreadd
  3 root    20   0    0     0    0 S 0.0  0.0 0:00.00 pool_workqueue_release
  4 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/R-rcu_g
  5 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/R-rcu_p
  6 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/R-slub_
  7 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/R-netns
 10 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/0:0H-events_highpri
 12 root    0 -20    0     0    0 I 0.0  0.0 0:00.00 kworker/R-mm_pe
 13 root    20   0    0     0    0 I 0.0  0.0 0:00.00 rcu_tasks_kthread
 14 root    20   0    0     0    0 I 0.0  0.0 0:00.00 rcu_tasks_rude_kthread

```

```

SQL statistics:
queries performed:
  read:                      191128
  write:                     54605
  other:                     27303
  total:                     273036
  transactions:             13651 (227.50 per sec.)
  queries:                  273036 (4550.25 per sec.)
  ignored errors:           1 (0.02 per sec.)
  reconnects:                0 (0.00 per sec.)

General statistics:
  total time:                60.0038s
  total number of events:    13651

Latency (ms):
  min:                       3.88
  avg:                        8.79
  max:                       64.05
  95th percentile:            17.32
  sum:                      119935.25

Threads fairness:
  events (avg/stddev):     6825.5000/9.50
  execution time (avg/stddev): 59.9676/0.00

```

root@nodo1:/home/abdiel# |

La prueba oltp\_read\_write muestra que el Core 2 ofrece un mayor rendimiento en términos de transacciones por segundo, reflejado en un mayor consumo de CPU por parte de MariaDB (102.0% frente a 66.2% en el Core 1). Sin embargo, este rendimiento se logra a costa de una mayor espera de E/S (16.0% frente a 12.2%), lo que puede afectar la latencia bajo alta carga. El Core 2 es ideal si se prioriza el procesamiento de transacciones, mientras que el Core 1 ofrece un rendimiento más equilibrado y eficiente en términos de latencia y consumo de recursos.

La prueba oltp\_read\_write muestra que el Core 2 procesa más transacciones por segundo, con un uso de CPU en MariaDB del 102%, frente al 66.2% en el Core 1. Sin embargo, este mayor rendimiento conlleva una mayor latencia debido a la espera de E/S, con un 16% en el Core 2 frente al 12.2% en el Core 1. Además, el Core 2 tiene un mayor uso de CPU para procesamiento activo (20% en modo usuario y 30.5% en modo sistema), mientras que el Core 1 mantiene un uso más equilibrado (4.4% us y 47.5% sy). En general, el Core 2 es ideal para maximizar el rendimiento transaccional, mientras que el Core 1 ofrece menor latencia y mayor eficiencia bajo cargas moderadas.

### **oltp\_update\_index**

En el contexto de la herramienta Sysbench, oltp\_update\_index es un tipo de prueba de rendimiento que evalúa la capacidad de una base de datos para manejar operaciones de actualización de índices en tablas con grandes volúmenes de datos. Esta prueba se enfoca en medir el impacto que tienen las actualizaciones de campos indexados en el rendimiento general del sistema y la base de datos.

#### **1 core**

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	461916	158208	S	48.2	5.7	4:55.91	mariadb
43	root	0	-20	0	0	0	I	4.0	0.0	0:07.70	kworker/0:1H-kblockd
1564	root	20	0	99364	13952	11264	S	2.0	0.2	0:00.48	sysbench
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:01.64	sshd
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.94	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:01.07	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.13	rcu_preempt
18	rt	0	0	0	0	0	S	0.0	0.0	0:00.04	migration/0
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0

```

SQL statistics:
  queries performed:
    read:                      0
    write:                     32347
    other:                     0
    total:                     32347
  transactions:               32347 (539.10 per sec.)
  queries:                   32347 (539.10 per sec.)
  ignored errors:            0 (0.00 per sec.)
  reconnects:                 0 (0.00 per sec.)

General statistics:
  total time:                60.0014s
  total number of events:    32347

Latency (ms):
  min:                       1.15
  avg:                       1.85
  max:                       14.32
  95th percentile:           2.52
  sum:                      59915.49

Threads fairness:
  events (avg/stddev):     32347.0000/0.00
  execution time (avg/stddev): 59.9155/0.00

```

root@nodo1:/home/abdiel# |

2 cores

top - 19:37:59 up 24 min, 2 users, load average: 0.65, 0.89, 0.63											
Tasks: 119 total, 1 running, 118 sleeping, 0 stopped, 0 zombie											
%Cpu(s): 7.3 us, 14.0 sy, 0.0 ni, 29.0 id, 24.7 wa, 0.0 hi, 24.9 si, 0.0 st											
MiB Mem : 7942.1 total, 7034.0 free, 738.7 used, 403.1 buff/cache											
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7203.4 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1345	mysql	20	0	1748096	487532	156928	S	50.0	6.0	7:19.92	mariadb
1570	root	20	0	165288	14080	11136	S	3.6	0.2	0:00.43	sysbench
50	root	0	-20	0	0	0	I	2.6	0.0	0:03.12	kworker/0:1H-kblockd
24	root	20	0	0	0	0	S	1.0	0.0	0:03.52	ksoftirqd/1
1439	root	0	-20	0	0	0	I	0.7	0.0	0:03.63	kworker/1:1H-kblockd
9	root	20	0	0	0	0	I	0.3	0.0	0:02.07	kworker/0:1-events
<b>1423</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5760</b>	<b>3712</b>	<b>R</b>	<b>0.3</b>	<b>0.1</b>	<b>0:01.57</b>	<b>top</b>
1480	root	20	0	0	0	0	I	0.3	0.0	0:01.55	kworker/1:30-events
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.97	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:01.58	kworker/u4:0-events_power_effi+
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread

```

SQL statistics:
  queries performed:
    read:                      0
    write:                     24693
    other:                     0
    total:                     24693
  transactions:              24693  (411.53 per sec.)
  queries:                  24693  (411.53 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                0      (0.00 per sec.)

General statistics:
  total time:                60.0026s
  total number of events:    24693

Latency (ms):
  min:                       1.57
  avg:                       4.86
  max:                       28.83
  95th percentile:           6.32
  sum:                      119909.46

Threads fairness:
  events (avg/stddev):     12346.5000/3.50
  execution time (avg/stddev): 59.9547/0.00

```

root@nodo1:/home/abdiel# |

La prueba `oltp_update_index` en el Core 1 muestra un rendimiento limitado por la velocidad del almacenamiento. El alto porcentaje de espera de E/S (33.3%) indica que el cuello de botella principal está en las operaciones de escritura y actualización de índices, lo que incrementa la latencia y reduce la capacidad de procesamiento de transacciones por segundo. Aunque la CPU no está saturada en modo usuario (1.2%), el alto uso en modo sistema (59.3%) sugiere que el kernel está manejando intensamente las operaciones de E/S.

La prueba `oltp_update_index` muestra que el Core 2 ofrece un mejor rendimiento general en comparación con el Core 1. El consumo de CPU por parte de MariaDB es ligeramente mayor (50.0% frente a 48.2%), indicando una mayor capacidad para procesar transacciones por segundo. Además, la espera de E/S en el Core 2 es del 24.7%, significativamente menor que el 33.3% del Core 1, lo que reduce la latencia y mejora la eficiencia del procesamiento.

La prueba `oltp_update_index` muestra que el Core 2 tiene un mejor rendimiento general en comparación con el Core 1. La latencia de ejecución es menor en el Core 2, con un 24.7% de espera de E/S frente al 33.3% en el Core 1, lo que indica una menor saturación del almacenamiento. Además, el Core 2 procesa más transacciones por segundo, reflejado en un consumo de CPU de MariaDB del 50% frente al 48.2% del Core 1. La distribución de la carga de CPU también es más equilibrada en el Core 2, con un 7.3% en modo usuario y 14.0% en modo sistema, mientras que el Core 1 muestra un desbalance con solo 1.2% en modo usuario y 59.3% en modo sistema. En general, el Core 2 es más eficiente para manejar actualizaciones de índices, con menor latencia y mejor aprovechamiento de recursos.

## oltp\_update\_non\_index

oltp\_update\_non\_index es una prueba de rendimiento diseñada para evaluar la capacidad de una base de datos para manejar operaciones de actualización en columnas no indexadas. A diferencia de oltp\_update\_index, esta prueba se enfoca en campos sin índices, lo que generalmente implica un mayor esfuerzo de procesamiento, ya que la base de datos debe escanear la tabla completa para localizar y actualizar los registros.

### 1 core

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	462300	158208	S	49.5	5.7	5:35.70	mariadb
43	root	0	-20	0	0	0	I	4.0	0.0	0:11.18	kworker/0:1H-kblockd
1573	root	20	0	99492	13824	11136	S	2.3	0.2	0:01.00	sysbench
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.94	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:01.09	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.13	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.05	migration/0
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
21	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs

```
SQL statistics:
  queries performed:
    read:                      0
    write:                     32506
    other:                     0
    total:                     32506
  transactions:              32506  (541.75 per sec.)
  queries:                   32506  (541.75 per sec.)
  ignored errors:            0    (0.00 per sec.)
  reconnects:                 0    (0.00 per sec.)

General statistics:
  total time:                60.0012s
  total number of events:     32506

Latency (ms):
  min:                        1.14
  avg:                         1.84
  max:                        25.24
  95th percentile:             2.43
  sum:                       59913.18

Threads fairness:
  events (avg/stddev):      32506.0000/0.00
  execution time (avg/stddev): 59.9132/0.00

root@nodo1:/home/abdiel# |
```

### 2 core

top - 19:39:50 up 26 min, 2 users, load average: 0.70, 0.87, 0.66											
Tasks: 119 total, 1 running, 118 sleeping, 0 stopped, 0 zombie											
%Cpu(s): 6.1 us, 14.6 sy, 0.0 ni, 29.8 id, 25.7 wa, 0.0 hi, 23.7 si, 0.0 st											
MiB Mem : 7942.1 total, 7032.0 free, 739.5 used, 404.4 buff/cache											
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7202.6 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1345	mysql	20	0	1748096	487916	156928	S	48.3	6.0	7:51.03	mariadb
1585	root	20	0	165544	14336	11264	S	3.7	0.2	0:00.32	sysbench
50	root	0	-20	0	0	0	I	2.7	0.0	0:04.78	kworker/0:1H-kblockd
24	root	20	0	0	0	0	S	1.0	0.0	0:04.36	ksoftirqd/1
59	root	20	0	0	0	0	I	1.0	0.0	0:00.73	kworker/u4:3-events_power_effi+
1439	root	0	-20	0	0	0	I	1.0	0.0	0:04.32	kworker/1:1H-kblockd
<b>1423</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5760</b>	<b>3712</b>	<b>R</b>	<b>0.7</b>	<b>0.1</b>	<b>0:01.77</b>	<b>top</b>
243	root	20	0	0	0	0	S	0.3	0.0	0:00.80	jbd2/dm-0-8
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.97	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
9	root	20	0	0	0	0	I	0.0	0.0	0:02.16	kworker/0:1-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:01.66	kworker/u4:0-events_power_effi+
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread

```
SQL statistics:
  queries performed:
    read:                      0
    write:                     21979
    other:                     0
    total:                     21979
  transactions:              21979  (366.28 per sec.)
  queries:                   21979  (366.28 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                 0      (0.00 per sec.)

General statistics:
  total time:                60.0056s
  total number of events:     21979

Latency (ms):
  min:                       1.76
  avg:                        5.46
  max:                      197.22
  95th percentile:           6.91
  sum:                      119923.63

Threads fairness:
  events (avg/stddev):     10989.5000/0.50
  execution time (avg/stddev): 59.9618/0.00

root@nodo1:/home/abdiel#
```

La prueba oltp\_update\_non\_index en el Core 1 muestra un rendimiento limitado por la velocidad del almacenamiento. El alto porcentaje de espera de E/S (31.7%) indica que el cuello de botella principal está en las operaciones de escritura y en los escaneos completos de la tabla para encontrar las filas a actualizar. Aunque la CPU no está saturada en modo usuario (1.2%), el alto uso en modo sistema (60.2%) sugiere que el kernel está manejando intensamente las operaciones de E/S.

La prueba oltp\_update\_non\_index muestra que el Core 2 ofrece un rendimiento superior al Core 1. El consumo de CPU por parte de MariaDB se mantiene estable (48.3% vs. 49.5%), pero la espera de E/S es menor en el Core 2 (25.7% frente a 31.7%), lo que reduce la latencia y permite un procesamiento más eficiente. Además, el menor uso de CPU en modo sistema (14.6% frente a 60.2%) indica una mejor gestión de las operaciones de actualización sin índices.

La prueba `oltp_update_non_index` muestra que el Core 2 supera al Core 1 en eficiencia y rendimiento. El tiempo de ejecución en Core 2 es menor, con una espera de E/S del 25.7%, frente al 31.7% en el Core 1, lo que indica una latencia reducida en las actualizaciones de columnas no indexadas. En términos de transacciones por segundo, ambos núcleos mantienen un rendimiento similar, con un consumo de CPU en MariaDB del 48.3% en Core 2 y 49.5% en Core 1. Sin embargo, la distribución del uso de CPU es más equilibrada en el Core 2, con 6.1% en modo usuario y 14.6% en modo sistema, mientras que el Core 1 muestra un desbalance con solo 1.2% en modo usuario y 60.2% en modo sistema. Además, el menor cuello de botella en el almacenamiento refuerza la superioridad del Core 2 para manejar cargas intensivas de actualizaciones sin índices.

### **`oltp_write_only`**

`oltp_write_only` es una prueba de rendimiento diseñada para evaluar la capacidad de una base de datos para manejar operaciones de escritura intensiva, como `INSERT`, `UPDATE` y `DELETE`, sin realizar lecturas. A diferencia de pruebas mixtas como `oltp_read_write`, esta prueba se enfoca exclusivamente en la carga generada por las transacciones de escritura.

#### **Core 1**

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	463068	158208	S	61.6	5.7	5:50.67	mariadb
1600	root	20	0	99492	13952	11264	S	6.0	0.2	0:00.66	sysbench
43	root	0	-20	0	0	0	I	4.0	0.0	0:12.39	kworker/0:1H-kblockd
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:01.82	sshd
1484	root	20	0	0	0	0	I	0.3	0.0	0:00.39	kworker/0:103-ata_sff
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.97	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:01.11	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.13	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.05	migration/0
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0

La prueba `oltp_write_only` en el Core 1 muestra un rendimiento considerablemente afectado por la velocidad del almacenamiento. El alto porcentaje de espera de E/S (16.4%) indica que las transacciones de escritura están limitadas por la capacidad del disco, lo que incrementa la latencia y reduce la eficiencia general. Aunque la CPU está activamente involucrada en modo sistema (63.4%), la baja utilización en modo usuario (1.4%) sugiere que el procesamiento de las transacciones está más limitado por la E/S que por la capacidad de cómputo.

```

SQL statistics:
  queries performed:
    read:                      0
    write:                     103476
    other:                     51738
    total:                    155214
  transactions:              25869 (431.14 per sec.)
  queries:                  155214 (2586.83 per sec.)
  ignored errors:            0 (0.00 per sec.)
  reconnects:                0 (0.00 per sec.)

General statistics:
  total time:                60.0006s
  total number of events:    25869

Latency (ms):
  min:                       1.58
  avg:                        2.32
  max:                       22.95
  95th percentile:           3.13
  sum:                      59904.92

Threads fairness:
  events (avg/stddev):     25869.0000/0.00
  execution time (avg/stddev): 59.9049/0.00

root@nodo1:/home/abdiel# |

```

2 cores

top - 19:41:57 up 28 min, 2 users, load average: 1.15, 0.95, 0.72										
Tasks: 121 total, 1 running, 120 sleeping, 0 stopped, 0 zombie										
%Cpu(s): 11.2 us, 28.0 sy, 0.0 ni, 24.6 id, 16.4 wa, 0.0 hi, 19.9 si, 0.0 st										
MiB Mem : 7942.1 total, 7027.4 free, 743.3 used, 405.2 buff/cache										
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7198.8 avail Mem										
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
1345	mysql	20	0	1748096	488428	157312	S	66.4	6.0	8:27.46 mariadb
1597	root	20	0	165544	14464	11136	S	20.6	0.2	0:03.18 sysbench
24	root	20	0	0	0	0	S	1.7	0.0	0:05.36 ksoftirqd/1
50	root	0	-20	0	0	0	I	1.7	0.0	0:06.36 kworker/0:1H-kblockd
1439	root	0	-20	0	0	0	I	1.3	0.0	0:05.14 kworker/1:1H-kblockd
11	root	20	0	0	0	0	I	0.3	0.0	0:01.83 kworker/u4:0-events_power_efficiency
<b>1423</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5760</b>	<b>3712</b>	<b>R</b>	<b>0.3</b>	<b>0.1</b>	<b>0:01.98 top</b>
1480	root	20	0	0	0	0	I	0.3	0.0	0:02.06 kworker/1:30-events
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.97 systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02 kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00 pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-rCU_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-rCU_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-netns
9	root	20	0	0	0	0	I	0.0	0.0	0:02.27 kworker/0:1-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/0:0H-events_highpri
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_tasks_rude_kthread

La prueba oltp\_write\_only muestra que el Core 2 ofrece un mejor rendimiento en comparación con el Core 1. El consumo de CPU por parte de MariaDB es mayor (66.4% frente a 61.6%), indicando una mayor capacidad para procesar transacciones por segundo. Además, el mayor uso de CPU en modo usuario (11.2%) sugiere que el Core 2 está manejando las operaciones de manera más eficiente y equilibrada, distribuyendo mejor la carga entre el modo usuario y el modo sistema.

Sin embargo, la espera de E/S sigue siendo un factor limitante en ambos núcleos, lo que sugiere que el almacenamiento es el principal cuello de botella. En general, el Core 2 es la mejor opción para

manejar cargas intensivas de escritura debido a su mayor capacidad de procesamiento y mejor distribución de la carga

```
SQL statistics:
  queries performed:
    read:                      0
    write:                     68248
    other:                     34124
    total:                    102372
  transactions:              17062  (284.34 per sec.)
  queries:                  102372 (1706.02 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                0      (0.00 per sec.)

General statistics:
  total time:                60.0056s
  total number of events:    17062

Latency (ms):
  min:                       2.25
  avg:                       7.03
  max:                      238.20
  95th percentile:           9.22
  sum:                      119919.76

Threads fairness:
  events (avg/stddev):     8531.0000/12.00
  execution time (avg/stddev): 59.9599/0.00

root@nodo1:/home/abdiel# |
```

La prueba oltp\_write\_only muestra que ambos núcleos presentan una latencia moderada, con un 16.4% de espera de E/S, lo que indica un cuello de botella persistente en el almacenamiento. Sin embargo, el Core 2 destaca por procesar más transacciones por segundo, con un consumo de CPU en MariaDB del 66.4%, frente al 61.6% del Core 1. Además, el Core 2 distribuye mejor la carga entre usuario y sistema, con un 11.2% de uso en modo usuario y 28.0% en modo sistema, mientras que el Core 1 muestra un desequilibrio, con solo 1.4% en modo usuario y 63.4% en modo sistema. En general, el Core 2 ofrece un rendimiento superior y una mejor eficiencia en la gestión de la carga, aunque el almacenamiento sigue siendo el principal factor limitante.

### **select\_random\_points**

select\_random\_points es una prueba de rendimiento diseñada para evaluar la capacidad de una base de datos para manejar consultas SELECT aleatorias basadas en claves primarias. Esta prueba mide la eficiencia de las consultas de lectura cuando se seleccionan puntos de datos específicos de manera aleatoria dentro de una tabla. Un buen rendimiento se refleja en una baja latencia, un alto número de consultas por segundo y un uso equilibrado de CPU y memoria.

### **Core 1**

top - 19:05:20 up 41 min, 2 users, load average: 0.44, 0.47, 0.40												
Tasks: 112 total, 1 running, 111 sleeping, 0 stopped, 0 zombie												
%Cpu(s): 26.9 us, 6.7 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 66.3 si, 0.0 st												
MiB Mem : 7942.4 total, 7047.1 free, 723.8 used, 405.1 buff/cache												
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7218.6 avail Mem												
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND	
1273	mysql	20	0	1297312	463196	158208	S	88.1	5.7	6:31.33	mariadb	
1610	root	20	0	99724	14444	11264	S	8.6	0.2	0:01.20	sysbench	
1119	abdiel	20	0	15128	7092	5120	S	2.0	0.1	0:08.48	sshd	
16	root	20	0	0	0	0	S	0.7	0.0	0:01.18	ksoftirqd/0	
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:01.91	sshd	
<b>1152</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5888</b>	<b>3712</b>	<b>R</b>	<b>0.3</b>	<b>0.1</b>	<b>0:02.43</b>	<b>top</b>	
1484	root	20	0	0	0	0	I	0.3	0.0	0:00.65	kworker/0:103-events	
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.97	systemd	
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd	
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release	
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g	
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p	
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub	
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns	
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe	
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread	
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread	
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread	
17	root	20	0	0	0	0	I	0.0	0.0	0:00.14	rcu_preempt	
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.05	migration/0	

```
ssSQL statistics:
  queries performed:
    read:                      33523
    write:                     0
    other:                     0
    total:                     33523
  transactions:              33523  (558.70 per sec.)
  queries:                   33523  (558.70 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                 0      (0.00 per sec.)

General statistics:
  total time:                60.0008s
  total number of events:     33523

Latency (ms):
  min:                        0.28
  avg:                        1.79
  max:                        10.19
  95th percentile:            4.10
  sum:                       59892.66

Threads fairness:
  events (avg/stddev):      33523.0000/0.00
  execution time (avg/stddev): 59.8927/0.00

root@nodo1:/home/abdiel# s|
```

La prueba select\_random\_points en el Core 1 muestra un rendimiento óptimo, con un alto número de transacciones por segundo y una baja latencia. El alto consumo de CPU en modo usuario y las interrupciones suaves sugieren que las consultas se están ejecutando rápidamente desde la memoria, sin cuellos de botella en el almacenamiento. Este rendimiento indica que la base de datos está bien optimizada para consultas de lectura aleatoria, siempre que la mayor parte de los datos se mantenga en la memoria caché.

## 2 cores

top - 19:43:54 up 30 min, 2 users, load average: 1.12, 0.98, 0.75										
Tasks: 121 total, 2 running, 119 sleeping, 0 stopped, 0 zombie										
%Cpu(s): 72.1 us, 23.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 4.4 si, 0.0 st										
MiB Mem : 7942.1 total, 7029.1 free, 741.1 used, 405.7 buff/cache										
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 7200.9 avail Mem										
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
1345	mysql	20	0	1748096	488940	157312	S	155.3	6.0	9:20.19 mariadb
1606	root	20	0	165800	14592	11264	S	44.0	0.2	0:06.70 sysbench
9	root	20	0	0	0	0	I	0.3	0.0	0:02.37 kworker/0:1-events
<b>1480</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>R</b>	<b>0.3</b>	<b>0.0</b>	<b>0:02.29 kworker/1:30-events</b>
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.98 systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02 kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00 pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-slab_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-netns
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:01.89 kworker/u4:0-flush-252:0
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:00.07 ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.52 rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.08 migration/0

```
SQL statistics:
  queries performed:
    read:                      1024524
    write:                     0
    other:                     0
    total:                     1024524
  transactions:              1024524 (17073.84 per sec.)
  queries:                   1024524 (17073.84 per sec.)
  ignored errors:            0      (0.00 per sec.)
  reconnects:                 0      (0.00 per sec.)

General statistics:
  total time:                60.0036s
  total number of events:     1024524

Latency (ms):
  min:                        0.07
  avg:                        0.12
  max:                        8.87
  95th percentile:            0.15
  sum:                       118962.06

Threads fairness:
  events (avg/stddev):      512262.0000/10892.00
  execution time (avg/stddev): 59.4810/0.01
root@nodo1:/home/abdiel# |
```

La prueba select\_random\_points muestra que el Core 2 ofrece un rendimiento significativamente superior en comparación con el Core 1. El consumo de CPU por parte de MariaDB es notablemente mayor (155.3% frente a 88.1%), indicando una mayor capacidad para procesar consultas por segundo. Además, el mayor uso de CPU en modo usuario (72.1%) sugiere que el Core 2 está ejecutando las consultas con mayor eficiencia y menor intervención del kernel.

La ausencia de espera de E/S en ambos núcleos indica que las lecturas se realizan completamente desde la memoria caché. En términos generales, el Core 2 es la mejor opción para manejar cargas intensivas de lecturas aleatorias.

La prueba select\_random\_points muestra que ambos núcleos presentan baja latencia (0.0% wa) y operan sin cuellos de botella en almacenamiento. Sin embargo, el Core 2 destaca por procesar

más transacciones por segundo, con un consumo de CPU en MariaDB del 155.3%, frente al 88.1% del Core 1. Además, el Core 2 distribuye mejor la carga entre el modo usuario (72.1%) y el modo sistema (23.5%), mientras que el Core 1 muestra un 26.9% en modo usuario y 6.7% en modo sistema. Esto indica que el Core 2 no solo maneja un mayor volumen de consultas, sino que lo hace de manera más eficiente, consolidándose como la mejor opción para cargas intensivas de lecturas aleatorias.

### **select\_random\_ranges**

**En el contexto de la herramienta Sysbench, select\_random\_ranges es una prueba de rendimiento diseñada para evaluar la capacidad de una base de datos para manejar consultas SELECT aleatorias que devuelven rangos de múltiples filas. A diferencia de select\_random\_points, que selecciona registros individuales, esta prueba mide la eficiencia cuando se consultan varios registros consecutivos en una sola operación.**

#### **1 core**

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1273	mysql	20	0	1297312	463196	158208	S	94.7	5.7	7:25.82	mariadb
1119	abdiel	20	0	15128	7092	5120	S	2.3	0.1	0:09.13	sshd
1614	root	20	0	99620	14080	11264	S	2.3	0.2	0:00.34	sysbench
<b>16</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>R</b>	<b>0.3</b>	<b>0.0</b>	<b>0:01.27</b>	<b>ksoftirqd/0</b>
1063	abdiel	20	0	15124	7088	5120	S	0.3	0.1	0:02.03	sshd
<b>1152</b>	<b>root</b>	<b>20</b>	<b>0</b>	<b>11908</b>	<b>5888</b>	<b>3712</b>	<b>R</b>	<b>0.3</b>	<b>0.1</b>	<b>0:02.63</b>	<b>top</b>
1484	root	20	0	0	0	0	I	0.3	0.0	0:01.13	kworker/0:103-events
1	root	20	0	22264	13424	9456	S	0.0	0.2	0:02.97	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
17	root	20	0	0	0	0	I	0.0	0.0	0:00.14	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.06	migration/0

La prueba select\_random\_ranges en el Core 1 muestra un rendimiento eficiente, con baja latencia y un alto número de transacciones por segundo. El alto consumo de CPU por parte de MariaDB (94.7%) y la ausencia de espera de E/S sugieren que las consultas de rangos aleatorios se están manejando completamente desde la memoria, sin cuellos de botella en el almacenamiento.

Este rendimiento indica que la base de datos está bien optimizada para consultas de rango siempre que los datos estén en caché. Sin embargo, el alto porcentaje de interrupciones suaves (65.7%) podría indicar una mayor carga del kernel, lo cual es importante monitorear si la carga aumenta.

```

SSQL statistics:
  queries performed:
    read:                      38931
    write:                     0
    other:                     0
    total:                     38931
  transactions:              38931 (648.82 per sec.)
  queries:                  38931 (648.82 per sec.)
  ignored errors:            0 (0.00 per sec.)
  reconnects:                0 (0.00 per sec.)

General statistics:
  total time:                60.00006s
  total number of events:    38931

Latency (ms):
  min:                       0.32
  avg:                       1.54
  max:                       13.82
  95th percentile:           3.75
  sum:                      59920.55

Threads fairness:
  events (avg/stddev):     38931.0000/0.00
  execution time (avg/stddev): 59.9205/0.00

```

root@nodo1:/home/abdiel# S

2 cores

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1345	mysql	20	0	1739900	488940	157312	S	156.8	6.0	10:46.93	mariadb
1622	root	20	0	165800	14592	11264	S	42.2	0.2	0:04.63	sysbench
11	root	20	0	0	0	0	I	0.3	0.0	0:01.95	kworker/u4:0-events_power_effi+
1480	root	20	0	0	0	0	I	0.3	0.0	0:02.52	kworker/1:30-events
1	root	20	0	22268	13256	9416	S	0.0	0.2	0:02.98	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthread
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
9	root	20	0	0	0	0	I	0.0	0.0	0:02.48	kworker/0:1-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
16	root	20	0	0	0	0	S	0.0	0.0	0:00.07	ksoftirqd/0
17	root	20	0	0	0	0	I	0.0	0.0	0:00.53	rcu_preempt
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.09	migration/0

La prueba select\_random\_ranges muestra que el Core 2 ofrece un rendimiento significativamente superior en comparación con el Core 1. El consumo de CPU por parte de MariaDB es notablemente mayor (156.8% frente a 94.7%), lo que indica una mayor capacidad para procesar transacciones por segundo. Además, el mayor uso de CPU en modo usuario (71.4%) sugiere que el Core 2 está ejecutando las consultas con mayor eficiencia y menor intervención del kernel.

La ausencia de espera de E/S en ambos núcleos confirma que las lecturas se realizan desde la memoria caché.

```

SQL statistics:
  queries performed:
    read:          1022226
    write:         0
    other:         0
    total:        1022226
  transactions:   1022226 (17036.27 per sec.)
  queries:       1022226 (17036.27 per sec.)
  ignored errors: 0      (0.00 per sec.)
  reconnects:    0      (0.00 per sec.)

General statistics:
  total time:      60.0019s
  total number of events: 1022226

Latency (ms):
  min:            0.07
  avg:           0.12
  max:           6.60
  95th percentile: 0.16
  sum:          118953.80

Threads fairness:
  events (avg/stddev): 511113.0000/11689.00
  execution time (avg/stddev): 59.4769/0.01

root@nodo1:/home/abdiel#

```

La prueba select\_random\_ranges muestra que ambos núcleos presentan baja latencia (0.0% wa) y operan sin cuellos de botella en el almacenamiento. Sin embargo, el Core 2 destaca por procesar más transacciones por segundo, con un consumo de CPU en MariaDB del 156.8%, frente al 94.7% del Core 1. Además, el Core 2 distribuye mejor la carga entre el modo usuario (71.4%) y el modo sistema (21.4%), mientras que el Core 1 muestra un 28.6% en modo usuario y 5.7% en modo sistema. Esto indica que el Core 2 no solo maneja un mayor volumen de consultas, sino que lo hace de manera más eficiente, consolidándose como la mejor opción para cargas intensivas de lecturas aleatorias de rangos.

### Métricas Consideradas durante la práctica.

- ◆ 1. Tiempo promedio de ejecución (avg latency).
- ◆ 2. Transacciones por segundo (transactions per second).
- ◆ 3. Utilización de CPU (%CPU): CPU usuario (us), CPU sistema (sy), Espera de E/S (wa), y Interrupciones suaves (si).
- ◆ 4. Velocidad de E/S (I/O): Espera de E/S.

Memoria: Total, Libre, Usada, Buffer/Cache.

- ◆ Swap: Total, Libre, Usada.
- ◆ Procesos principales: MariaDB (mariadb), Sysbench y Kworker.

### **Conclusión personal.**

Las pruebas de benchmarking realizadas, como Bulk\_insert, oltp\_delete, oltp\_insert, oltp\_point\_select, oltp\_read\_only, oltp\_read\_write, oltp\_update\_index, oltp\_update\_non\_index, oltp\_write\_only, select\_random\_points y select\_random\_ranges, proporcionaron una visión detallada del comportamiento del clúster bajo diferentes escenarios. Los resultados demostraron que la adición de un tercer nodo mejoró significativamente la capacidad de procesamiento y la resiliencia del sistema, permitiendo manejar un mayor número de transacciones por segundo y reduciendo la latencia promedio en la mayoría de las pruebas.

El análisis comparativo entre uno y dos cores reveló que el uso de múltiples núcleos optimiza la distribución de la carga, mejorando el rendimiento general sin aumentar significativamente el consumo de recursos. Las pruebas oltp\_read\_only y select\_random\_points destacaron por su eficiencia, manteniendo baja latencia y alto rendimiento, mientras que las pruebas de escritura intensiva, como oltp\_write\_only y bulk\_insert, mostraron mejoras notables con el tercer nodo.

En conclusión, la implementación de un clúster de bases de datos MariaDB no solo garantiza la disponibilidad y la integridad de los datos mediante la replicación, sino que también proporciona escalabilidad y un mejor rendimiento bajo cargas crecientes. Este estudio demuestra que la adición de nodos puede mejorar la eficiencia del sistema, siempre que se optimicen adecuadamente los recursos de hardware y la configuración de la base de datos. La documentación detallada de los resultados permite comprender los beneficios y desafíos de la escalabilidad en un entorno real, sirviendo como referencia para futuras implementaciones y optimizaciones.