

Rubrica 5: Sigue el tutorial Cassandra

SOs:



HP-VICTUS-15
Victus by HP Laptop 16-d1014ns

Cambiar el nombre

Especificaciones del dispositivo

Nombre del dispositivo: HP-VICTUS-15

Procesador: 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz

RAM instalada: 16.0 GB (15.7 GB usable)

Identificador de dispositivo: 4FB312CA-A29A-4ED5-BE63-0D4CE9B65E59

Id. del producto: 00330-80000-00000-AA009

Tipo de sistema: Sistema operativo de 64 bits, procesador basado en x64

Lápiz y entrada táctil: La entrada táctil o manuscrita no está disponible para esta pantalla

Vínculos relacionados: Dominio o grupo de trabajo | Protección del sistema | Configuración avanzada de la red

Especificaciones de Windows

Edición: Windows 11 Pro

Versión: 22H2

Instalado el: 23/11/2022

versión del sistema operativo: 22621.1344

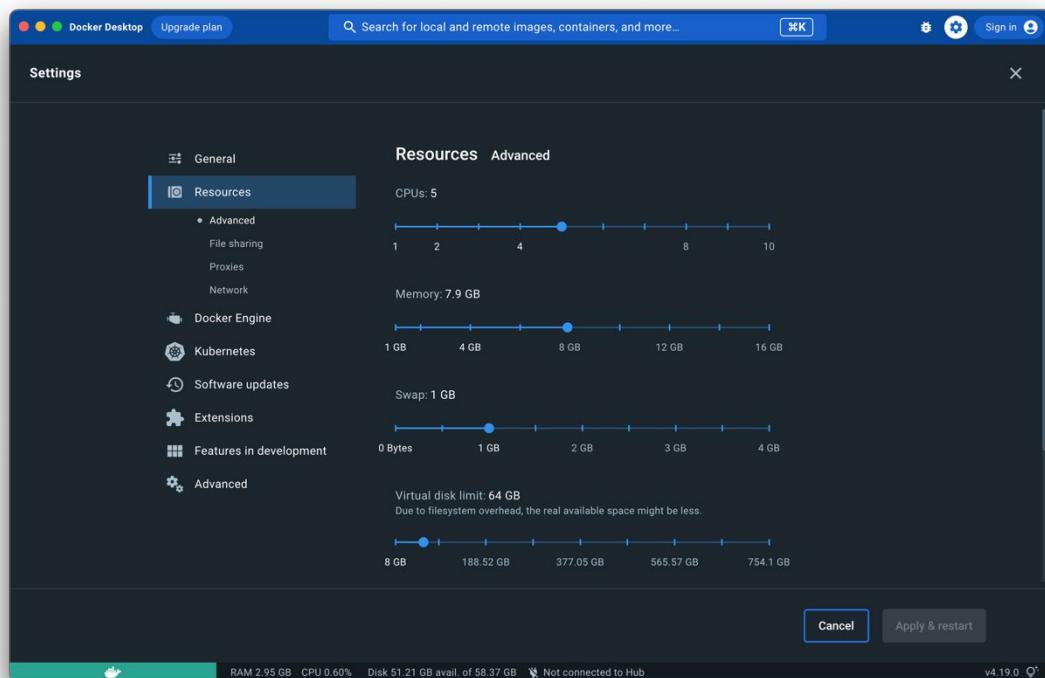
Experiencia: Windows Feature Experience Pack 1000.22639.1000.0

Contrato de servicios de Microsoft: Términos de licencia del software de Microsoft

Docker:

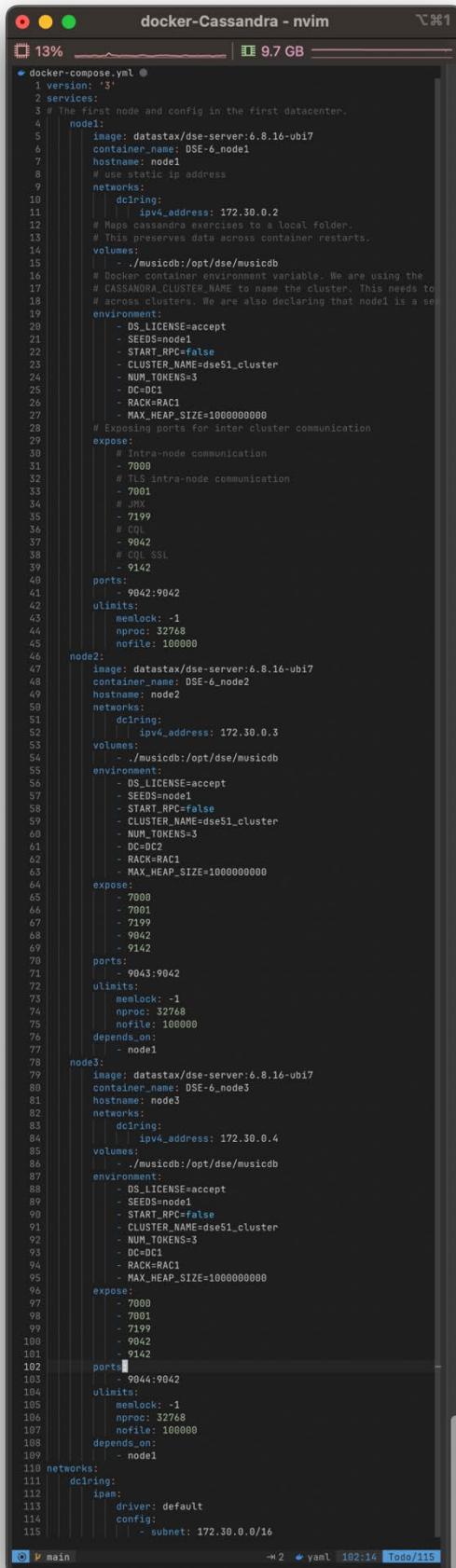


1. Nos aseguramos de que la configuración de Docker cumple con los requisitos del tutorial:



2. Creamos un directorio para almacenar el archivo de configuración:

3. Creamos el archivo de configuración de Docker Compose:



The screenshot shows a Mac OS X terminal window with a title bar "docker-Cassandra - nvim". The nvim editor is displaying the contents of a file named "docker-compose.yml". The code in the file defines a multi-node Cassandra cluster using Docker Compose. It includes three service definitions: "node1", "node2", and "node3". Each service uses the "datastax/dse-server:6.8.16-ubi7" image and has a static IP address assigned via the "dc1ring" network. The services are configured to use the same volumes, environment variables, and port mappings. The "node1" service is designated as the seed node ("SEEDS:node1"). The "node2" and "node3" services depend on "node1". The "node3" service has a specific IP address of "172.30.0.4". The "networks" section at the bottom defines the "dc1ring" network with a subnet of "172.30.0.0/16".

```
version: '3'
services:
  # The first node and config in the first datacenter.
  node1:
    image: datastax/dse-server:6.8.16-ubi7
    container_name: DSE-6_node1
    hostname: node1
    # use static ip address
    networks:
      dc1ring:
        ipv4_address: 172.30.0.2
    # Maps cassandra exercises to a local folder.
    # This preserves data across container restarts.
    volumes:
      - ./musicdb:/opt/dse/musicdb
    # Docker container environment variable. We are using the
    # CASSANDRA_CLUSTER_NAME to name the cluster. This needs to
    # across clusters. We are also declaring that node1 is a seed
    # node.
    environment:
      - DS_LICENSE=accept
      - SEEDS:node1
      - START_RPC=false
      - CLUSTER_NAME=dse51_cluster
      - NUM_TOKENS=3
      - DC=DC1
      - RACK=RAC1
      - MAX_HEAP_SIZE=1000000000
    # Exposing ports for inter cluster communication
    expose:
      # Intra-node communication
      - 7000
      # TLS intra-node communication
      - 7001
      # JMX
      - 7199
      # CQL
      - 9042
      # CQL SSL
      - 9142
    ports:
      - 9042:9042
    ulimits:
      memlock: -1
      nproc: 32768
      nofile: 100000
  node2:
    image: datastax/dse-server:6.8.16-ubi7
    container_name: DSE-6_node2
    hostname: node2
    networks:
      dc1ring:
        ipv4_address: 172.30.0.3
    volumes:
      - ./musicdb:/opt/dse/musicdb
    environment:
      - DS_LICENSE=accept
      - SEEDS:node1
      - START_RPC=false
      - CLUSTER_NAME=dse51_cluster
      - NUM_TOKENS=3
      - DC=DC2
      - RACK=RAC1
      - MAX_HEAP_SIZE=1000000000
    expose:
      - 7000
      - 7001
      - 7199
      - 9042
      - 9142
    ports:
      - 9043:9042
    ulimits:
      memlock: -1
      nproc: 32768
      nofile: 100000
    depends_on:
      - node1
  node3:
    image: datastax/dse-server:6.8.16-ubi7
    container_name: DSE-6_node3
    hostname: node3
    networks:
      dc1ring:
        ipv4_address: 172.30.0.4
    volumes:
      - ./musicdb:/opt/dse/musicdb
    environment:
      - DS_LICENSE=accept
      - SEEDS:node1
      - START_RPC=false
      - CLUSTER_NAME=dse51_cluster
      - NUM_TOKENS=3
      - DC=DC1
      - RACK=RAC1
      - MAX_HEAP_SIZE=1000000000
    expose:
      - 7000
      - 7001
      - 7199
      - 9042
      - 9142
    ports:
      - 9044:9042
    ulimits:
      memlock: -1
      nproc: 32768
      nofile: 100000
    depends_on:
      - node1
networks:
  dc1ring:
    ipam:
      driver: default
      config:
        - subnet: 172.30.0.0/16
```

4. Nos aseguramos de que se haya creado:



5. Levantamos el contenedor:

```
docker compose -f docker-compose.yml up
[+] Running 19/19
  ✓ node3 Pulled
  ✓ node1 16 layers [██████████████████]    0B/0B      Pulled   69.7s
    ✓ c13bd28f35dc Pull complete             1.7s
    ✓ 0dc69daaa449 Pull complete             1.7s
    ✓ 755d653b6228 Pull complete             8.9s
    ✓ a5ea20cc493e Pull complete             9.0s
    ✓ 4bf873279014 Pull complete             9.0s
    ✓ 44fde88c4818 Pull complete             9.0s
    ✓ f4e3008bfe4a Pull complete             9.1s
    ✓ ff9002066711 Pull complete             9.1s
    ✓ aa5021e1dd75 Pull complete             47.8s
    ✓ 0b4e5582d536 Pull complete             47.9s
    ✓ d3dcde48834f Pull complete             47.9s
    ✓ 54c750f038bd Pull complete             47.9s
    ✓ 5c0d7d13972c Pull complete             48.0s
    ✓ 93b60e864ca0 Pull complete             48.0s
    ✓ 8f8472b3ed49 Pull complete             67.1s
    ✓ 1a45cbded976 Pull complete             67.2s
  ✓ node2 Pulled
[+] Running 2/2
  ✓ Network docker-cassandra_dc1ring

  Created 0.0s
: Container DSE-6_node1

[+] Running 7/3
  ✓ Network docker-cassandra_dc1ring
```

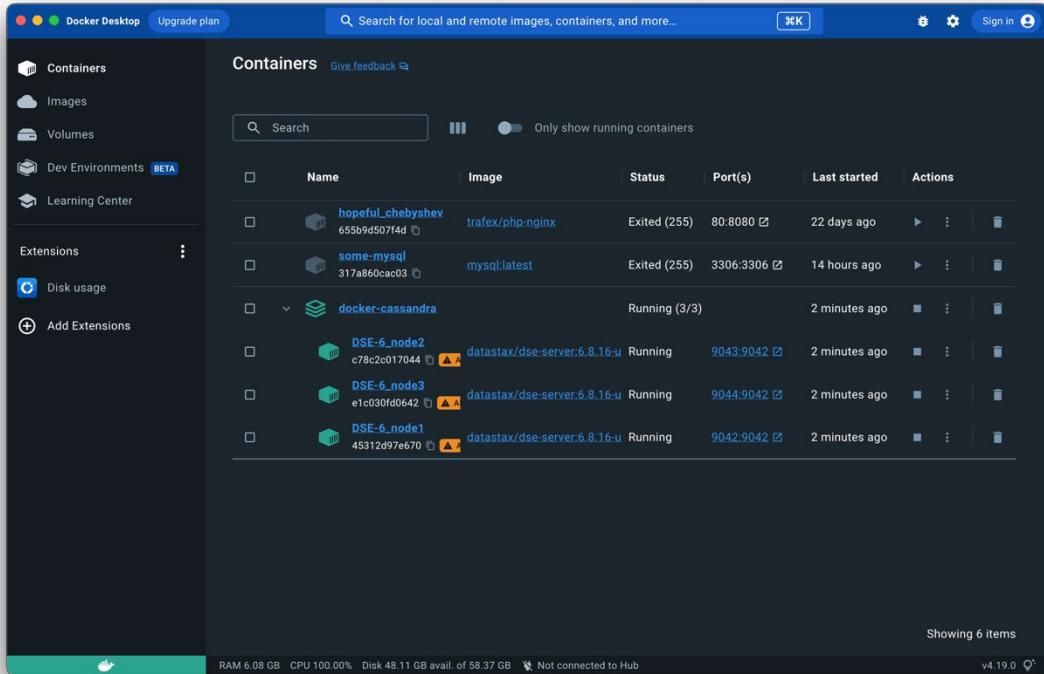
```

docker compose -f docker-compose.yml up
Created 0.0s
✓ Container DSE-6_node1
    Created 0.3s
    ! node1 The requested image's platform (linux/amd64) does not match the detected host platform (linux/arm64/v8) and no specific platform was requested 0.0s
    ✓ Container DSE-6_node2
        Created 0.1s
        ✓ Container DSE-6_node3
            Created 0.1s
            ! node2 The requested image's platform (linux/amd64) does not match the detected host platform (linux/arm64/v8) and no specific platform was requested 0.0s
            ! node3 The requested image's platform (linux/amd64) does not match the detected host platform (linux/arm64/v8) and no specific platform was requested 0.0s
Attaching to DSE-6_node1, DSE-6_node2, DSE-6_node3
DSE-6_node1 | Applying changes to /opt/dse/resources/cassandra/conf/cassandra.yaml ...
DSE-6_node1 | done.
DSE-6_node1 | Applying changes to /opt/dse/resources/cassandra/conf/cassandra-rackdc.properties ...
DSE-6_node1 | done.
DSE-6_node1 | Running dse cassandra -f -R
DSE-6_node3 | Applying changes to /opt/dse/resources/cassandra/conf/cassandra.yaml ...

```

6. Comprobamos que se ha levantado:

CONTAINER ID	IMAGE	COMMAND	CREATE	NAMES
D	STATUS	PORTS		
c78c2c017044	datastax/dse-server:6.8.16-ubi7	"/entrypoint.sh dse ..."	6 minutes ago	DSE-6_node2
e1c030fd0642	datastax/dse-server:6.8.16-ubi7	"/entrypoint.sh dse ..."	6 minutes ago	DSE-6_node3
45312d97e670	datastax/dse-server:6.8.16-ubi7	"/entrypoint.sh dse ..."	6 minutes ago	DSE-6_node1

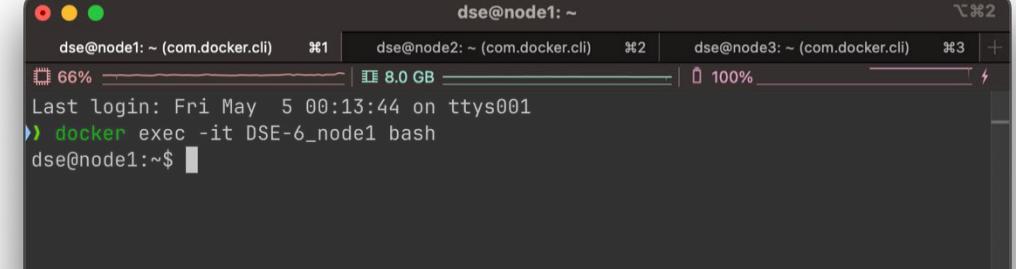


En este caso, al seguir un tutorial desde un ordenador con procesador ARM hay que desactivar esta característica para que el nodo 2 no se caiga con código de salida 3.

```
DSE-6_node2 | Running dse cassandra -f -R
DSE-6_node2 exited with code 3
DSE-6_node2 | Applying changes to /opt/dse/resources/cassandra.yaml ...
DSE-6_node2 | done.
DSE-6_node2 | Applying changes to /opt/dse/resources/cassandra-rackdc.properties ...
DSE-6_node2 | done.
DSE-6_node2 | Running dse cassandra -f -R
DSE-6_node2 exited with code 3
invalid character 's' looking for beginning of value
```

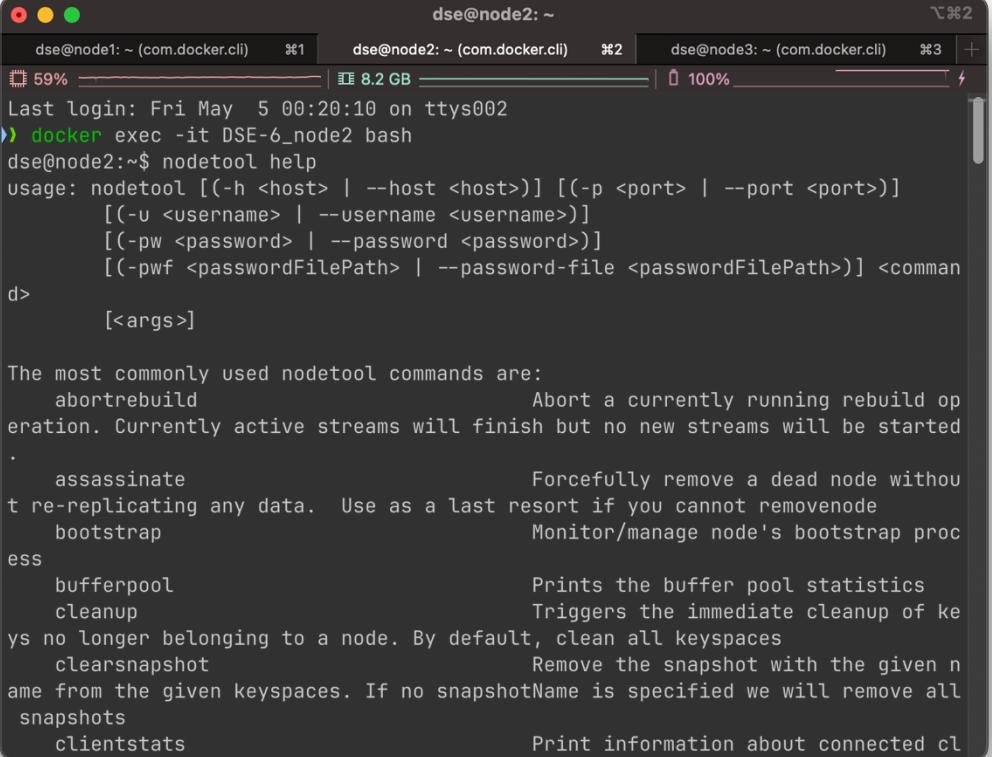
Use Rosetta for x86/amd64 emulation on Apple Silicon
Turns on Rosetta to accelerate x86/amd64 binary emulation on Apple Silicon. Note - you must have the Virtualization Framework enabled (via the toggle on the General panel).

7. Nos conectamos a los tres nodos con los comandos (hay un nodo en cada pestaña del terminal)



```
dse@node1: ~ (com.docker.cli) #1 dse@node2: ~ (com.docker.cli) #2 dse@node3: ~ (com.docker.cli) #3 +  
>Last login: Fri May  5 00:13:44 on ttys001  
↳ docker exec -it DSE-6_node1 bash  
dse@node1:~$
```

8. Comprobamos la conexión corriendo el comando ‘nodetool help’



```
dse@node2: ~ (com.docker.cli) #1 dse@node2: ~ (com.docker.cli) #2 dse@node3: ~ (com.docker.cli) #3 +  
Last login: Fri May  5 00:20:10 on ttys002  
↳ docker exec -it DSE-6_node2 bash  
dse@node2:~$ nodetool help  
usage: nodetool [(-h <host> | --host <host>)] [(-p <port> | --port <port>)]  
      [(-u <username> | --username <username>)]  
      [(-pw <password> | --password <password>)]  
      [(-pwf <passwordFilePath> | --password-file <passwordFilePath>)] <command>  
      [<args>]  
  
The most commonly used nodetool commands are:  
  abortrebuild          Abort a currently running rebuild operation. Currently active streams will finish but no new streams will be started.  
  assassinate           Forcefully remove a dead node without re-replicating any data. Use as a last resort if you cannot removenode.  
  bootstrap             Monitor/manage node's bootstrap process.  
  bufferpool            Prints the buffer pool statistics.  
  cleanup               Triggers the immediate cleanup of keyspaces no longer belonging to a node. By default, clean all keyspaces.  
  clearsnapshot         Remove the snapshot with the given name from the given keyspaces. If no snapshotName is specified we will remove all snapshots.  
  clientstats           Print information about connected clients.
```

9. Lanzamos el comando nodetool status:

```

alberto@MBP-de-Alberto:~/gitSalesianos/BBDD2223/docker-Cassandra ..ker-Cassandra (~sh) ● %1 ..ker-Cassandra (~sh) #2 + 11% 9.2 GB 402 kB↓ A.M. A. 45 kB↑ 100%
) docker exec -it DSE-6_node2 bash
dse@node2:~$ nodetool status
Datacenter: DC1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving/Stopped
-- Address Load Tokens Owns (effective) Host ID Rack
DS 172.30.0.2 0 bytes 3 100.0% cbb4f504-fe16-42c1-8532-ef0a8a76823e RAC1
Datacenter: DC2
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving/Stopped
-- Address Load Tokens Owns (effective) Host ID Rack
UJ 172.30.0.3 142.78 KiB 3 ? 8a8bb07c-02a6-4eeb-86e7-e6129510a01f RAC1

dse@node2:~$ 
) docker exec -it DSE-6_node2 bash
dse@node2:~$ nodetool status
nodetool: Failed to connect to '127.0.0.1:7199' - ConnectException: 'Connection refused (Connection refused)'.
dse@node2:~$ nodetool status
nodetool: Failed to connect to '127.0.0.1:7199' - ConnectException: 'Connection refused (Connection refused)'.
dse@node2:~$ exit
exit
) docker compose down
no configuration file provided: not found
) z cassa
) docker compose down
[+] Running 4/4
✓ Container DSE-6_node3 Removed 10.5s
✓ Container DSE-6_node2 Removed 10.4s
✓ Container DSE-6_node1 Removed 10.6s
✓ Network docker-cassandra_dc1ring Removed 0.0s
)

```

Empieza a dar problemas por la arquitectura, así que decido eliminar los contenedores y pasar a otro equipo con procesador Intel:

```

docker
) docker network prune
WARNING! This will remove all custom networks not used by at least one container.
Are you sure you want to continue? [y/N] y
) docker compose -f docker-compose.yml up
[+] Running 12/19
: node1 16 layers [██████████] 411.1MB/2.342GB Pulling 13.7s
✓ c13bd28f35dc Pull complete 2.0s
✓ 0dc69daaa449 Pull complete 2.0s
⌘ 755dd53b6228 Extracting 193.3MB/193.3MB 11.2s
✓ a5ea20cc493e Download complete 1.4s
✓ 4bfde88c4818 Download complete 2.2s
✓ f4e3008bfe4a Download complete
✓ ff9002066711 Download complete
" a5021e1dd75 Downloading 201.6MB/808.4MB
✓ 0b4e5582d536 Download complete 3.0s
✓ d3dcde48834f Download complete 11.2s
✓ 54c750f038bd Download complete 6.0s
⌘ 5c0d7d13972c Verifying Checksum 6.7s
✓ 93b60e864ca0 Download complete 7.9s
" 8f8472b3ed49 Downloading 16.19MB/1.34GB
✓ 1a45cbded976 Download complete 11.2s
: node2 Pulling 11.2s
: node3 Pulling 8.6s

```

Volvemos a lanzar nodetool status y ahora si que no da problemas:

```
cd /mnt/d/Salesianos/BBDD2223/
docker exec -it DSE-6_node1 bash
dse@node1:~$ nodetool status
Datacenter: DC1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving/Stopped
-- Address      Load      Tokens      Owns (effective)  Host ID
   Rack
UN 172.30.0.4  129.54 KiB  3          90.5%           c634d73e-a1a7-40a8-bc99-45d2a3f0
95e5  RAC1
UN 172.30.0.2  136.38 KiB  3          54.0%           8b7f9633-92f9-47cb-bab2-f3b62d36
8892  RAC1
Datacenter: DC2
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving/Stopped
-- Address      Load      Tokens      Owns (effective)  Host ID
   Rack
UN 172.30.0.3  119.46 KiB  3          55.6%           8d5356d5-6970-4c5a-b2d9-0866cb8a
dea9  RAC1
dse@node1:~$
```

The terminal window shows the execution of the 'nodetool status' command on a DSE node. The output indicates two nodes in Datacenter DC1 (RAC1 and RAC2) and one node in Datacenter DC2 (RAC1). All nodes are in the 'Normal' state. The Docker Desktop interface shows several containers running, including MySQL and Docker images.

10. Ahora lanzamos el comando cqlsh para empezar a trabajar en Cassandra:

```
dse@node1:~$ cqlsh
Connected to dse51_cluster at 127.0.0.1:9042.
[cqlsh 6.8.0 | DSE 6.8.16 | CQL spec 3.4.5 | DSE protocol v2]
Use HELP for help.
cqlsh> █
```

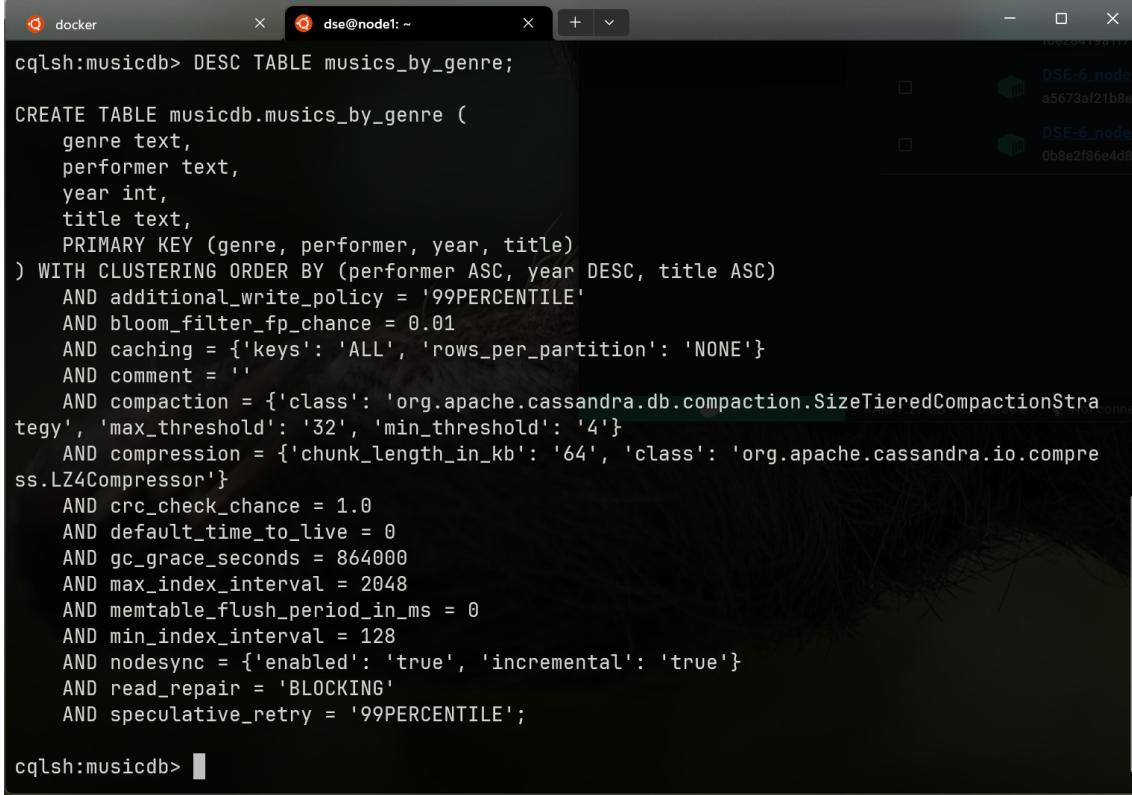
11. Consultamos las bases de datos (keyspaces)

```
cqlsh> DESC keyspaces;
system_virtual_schema  system_schema  dse_leases          system_traces
dse_system_local        system_auth    system_backups    dse_perf
dse_security            system_views   dse_insights     dse_insights_local
solr_admin              system        system_distributed dse_system
```

12. Creamos una base de datos 'musicDB', la usamos y creamos una tabla:

```
cqlsh> CREATE KEYSPACE musicDb WITH replication = {'class': 'SimpleStrategy', 'replication_factor' : '3'};
cqlsh> USE musicdb ;
cqlsh:musicdb> CREATE TABLE musics_by_genre (
    ... genre VARCHAR,
    ... performer VARCHAR,
    ... year INT,
    ... title VARCHAR,
MARY KEY    ... PRIMARY KEY ((genre), performer, year, title)
    ... ) WITH CLUSTERING ORDER BY (performer ASC, year DESC, title ASC);
```

13. Comprobamos que se ha creado:



```
cqlsh:musicdb> DESC TABLE musics_by_genre;

CREATE TABLE musicdb.musics_by_genre (
    genre text,
    performer text,
    year int,
    title text,
    PRIMARY KEY (genre, performer, year, title)
) WITH CLUSTERING ORDER BY (performer ASC, year DESC, title ASC)
    AND additional_write_policy = '99PERCENTILE'
    AND 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 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 nodesync = {'enabled': 'true', 'incremental': 'true'}
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99PERCENTILE';

cqlsh:musicdb>
```

14. Insertamos datos en la tabla:

```
cqlsh:musicdb> INSERT INTO musics_by_genre (genre, performer, year, title) VALUES ('Rock',
    'Nirvana', 1991, 'Smells Like Teen Spirit');
cqlsh:musicdb>
```

15. Ahora vamos a jugar con la consistencia del sistema, vamos a tirar dos de los nodos y a bajar la consistencia a 1 para ver si el sistema responde, desde un nuevo terminal:

```
› docker stop DSE-6_node2
DSE-6_node2
[ ] ➜ ~
docker stop DSE-6_node3
DSE-6_node1
f0e28419a1f7  0b8e2f86e4d8  DSE-6_node3  --      16 minutes, datastax/dse-server:6.8.16-u
[ ] ➜ took 11s ✘ at 01:31:08 ○
```

Docker Desktop interface showing three running containers:

- DSE-6_node1**: Running (datastax/dse-server:6.8.16-ub) 9042:9042, 18 minutes ago
- DSE-6_node2**: Exited (137) 9043:9042, 18 minutes ago
- DSE-6_node3**: Exited (137) 9044:9042, 18 minutes ago

RAM 3.67 GB CPU 2.01% Not connected to Hub v4.19.0

```

> docker stop DSE-6_node2
DSE-6_node2
> docker stop DSE-6_node3
DSE-6_node3
  
```

took 11s at 01:33:07

```

cqlsh:musicdb> CONSISTENCY ALL;
Consistency level set to ALL.
cqlsh:musicdb> SELECT * FROM musics_by_genre WHERE genre='Rock';
NoHostAvailable:
cqlsh:musicdb> CONSISTENCY ONE
      ; <enter>
cqlsh:musicdb> CONSISTENCY ONE ;
Consistency level set to ONE.
cqlsh:musicdb> SELECT * FROM musics_by_genre WHERE genre='Rock';

genre | performer | year | title
-----+-----+-----+
  Rock |    Nirvana | 1991 | Smells Like Teen Spirit

(1 rows)
cqlsh:musicdb> 
  
```

Y eso ha sido todo:

```

cqlsh:musicdb> EXIT
dse@node1:~$ exit
exit
> docker compose down
no configuration file provided: not found
> cd docker-Cassandra/
> docker compose down
[+] Running 4/4
  ✓ Container DSE-6_node2           Removed          0.1s
  ✓ Container DSE-6_node3           Removed          0.1s
  ✓ Container DSE-6_node1           Removed         10.8s
  ✓ Network docker-cassandra_dclring Removed          0.2s
  
```

/mnt/d/Sa/BBDD2223/docker-Cassandra on main