



SISTEMAS DE ACTA DISPONIBILIDAD

ÍNDICE

MI ELECCIÓN DE CASSANDRA COMO BD	3
¿QUÉ ES UNA BD CASSANDRA?	4
CARACTERÍSTICAS SOBRE CASSANDRA	5
CONCEPTOS ACERCA DE CASSANDRA	6
¿PARA QUÉ SE USA CASSANDRA?	7
¿QUÉ EMPRESAS LA USAN EN EL MERCADO?	7
¿QUÉ USO REAL SE LE PODRÍA DAR?	8
INSTALACIÓN Y ADMINISTRACIÓN	8
PASOS A SEGUIR PARA INSTALARLO	8
PARADA Y ARRANQUE DE CASSANDRA	11
UBICACIÓN DE ARCHIVOS GESTIÓN POR DEFECTO	12
CONEXIONES, ALMACENAMIENTO, ENTORNO, VARIABLES, PUERTOS...	15
Conexiones	15
Almacenamiento	20
Entornos y variables	23
Puertos	25
Nomenclatura de administración desde cli del sistema	26
HERRAMIENTA GRÁFICA Y EXPLICACIÓN DE LA ARQUITECTURA USADA	28
EXPLICACIÓN DE ARQUITECTURA MONTADA	29
CREACIÓN DE USUARIOS Y PERMISOS	35
AUTENTICACIÓN Y AUTORIZACIÓN	35
CREACIÓN DE USUARIOS	36
Crear un usuario	36
Ver usuarios creados	37
Modificar un usuario	37
Eliminar un usuario	38
ASIGNACIÓN DE PERMISOS	38
Otorgar permisos	38
Revocar permisos	39
Ver permisos	39
SEGURIDAD. DIRECTIVAS RELEVANTES DEL SGBD. HERRAMIENTAS DE BACKUPs.	40
AUTENTICACIÓN Y AUTORIZACIÓN	40
PERMISOS	41
CIFRADO DE DATOS	43
Cifrado de datos en tránsito	43
Cifrado de datos en reposo	44
¿Para qué sirve?	44
AUDITORÍAS	47
DIRECTIVAS RELEVANTES DEL SGBD	48
FIREWALL	50

HERRAMIENTAS DE BACKUPS.	53
Handy Backup	53
HERRAMIENTA DE SINCRONIZACIÓN DE DIRECTORIOS	60
Syncthing	60
MONITORIZACIÓN & LOGS	69
PARÁMETROS	69
Cqlsh	69
Logs importantes	70
Parámetros Monitoreo en Tiempo Real	70
Rutas de logs	71
HERRAMIENTA DE MONITORIZACIÓN (DATADOG)	71
PASOS PARA PODER USAR DATADOG	72
Alerta datadog	86
SCRIPTS AUTOMATIZACIÓN DE TAREAS	88
SCRIPT MONITOREO (ESTADO DE LOS NODOS DEL CLÚSTER CASSANDRA)	88
MIGRACIÓN BD	90
COPIA DE SEGURIDAD BD (EN BASH)	92
RESTAURACIÓN COPIA BD (EN BASH)	93
INVENTARIO DE EQUIPO	94
ALERTA DISCORD MONITORIZACIÓN SISTEMA (NODOS DE CASSANDRA)	95
OPTIMIZACIÓN DEL RENDIMIENTO	96
INFORMACIÓN IMPORTANTE ACERCA DE ÍNDICES CASSANDRA	96
PORQUE NO SE RECOMIENDA EL USO DE ÍNDICES SECUNDARIOS	96
CASO PRÁCTICO (CLÚSTER EN GOOGLE CLOUD PLATFORM Y MONITORIZACIÓN DE RECURSOS CON PROMETHEUS, FLUENTBIT Y GRAFANA)	97
INTRODUCCIÓN	97
¿QUÉ ES Y PARA QUÉ SIRVE GOOGLE CLOUD Y LAS HERRAMIENTAS POR DEFECTO MENCIONADAS ANTERIORMENTE ?	97
PASOS A SEGUIR PARA DESPLEGAR UN CLÚSTER DE CASSANDRA EN GOOGLE CLOUD	98
CONCLUSIÓN	109
WEBGRAFÍA	110
INFORMACIÓN ACERCA DE CASSANDRA	110
COMO INSTALAR Y CONFIGURACIONES CASSANDRA	110
CONEXIONES REMOTAS CASSANDRA	110
PASOS PARA INSTALAR E IMPLEMENTAR DATADOG	110

MI ELECCIÓN DE CASSANDRA COMO BD

He elegido **Cassandra** porque es una base de datos altamente escalable y distribuida que me permite gestionar grandes volúmenes de datos de manera eficiente, asegurando alta

disponibilidad y tolerancia a fallos. Además de su capacidad para manejar datos en tiempo real y su flexibilidad para adaptarse a proyectos de **Big Data** como análisis, recomendación y monitoreo, la hacen ideal para los casos de uso que necesito desarrollar.

¿QUÉ ES UNA BD CASSANDRA?

Apache Cassandra es un sistema de gestión de bases de datos **NoSQL** de **código abierto**, **diseñado para manejar grandes volúmenes de datos distribuidos de forma descentralizada**. Se utiliza principalmente cuando es necesario **garantizar una escalabilidad incluso cuando es necesario de forma masiva, alta disponibilidad y tolerancia a fallos** en entornos donde los datos deben estar siempre disponibles.

Cuenta con un lenguaje de consulta propio, denominado **Cassandra Query Language (CQL)**, que es muy **parecido al SQL**, pero preferido por los desarrolladores por estar hecho a la medida de las características especiales de Cassandra. Además esta **cuenta con un enfoque redundante, lo que reduce mucho la probabilidad de fallo**.

Esta base de datos nos proporciona **tolerancia a particiones y disponibilidad a cambio** de ser eventualmente **consistente**. Ese nivel de consistencia puede ser configurado según le interese al usuario o empresa que la use.

Es conocida también por proporcionar **escalabilidad de forma horizontal**; es decir nos permite añadir nuevos nodos. Aunque también permite una **escalabilidad lineal**, escalar linealmente por ejemplo es si al tener 2 nodos se puede soportar 100.000 operaciones por segundo, pues si añades 2 nodos más se soportará 200.000 operaciones por segundo.

También combina propiedades de una base de datos clave-valor y una orientada a columnas.

Una de sus características más destacadas es su arquitectura basada en un modelo **peer-to-peer**, donde todos los nodos son iguales y no existe un nodo maestro central. Esto elimina puntos únicos de fallo y permite que los nodos se comuniquen entre sí directamente, distribuyendo la carga y asegurando una alta disponibilidad. También facilita la replicación de datos de forma automática entre nodos, garantizando que los datos estén accesibles incluso si algunos nodos fallan.

CARACTERÍSTICAS SOBRE CASSANDRA

Característica	Descripción
Modelo de Datos	Basado en clave-valor y familias de columnas, con soporte para datos estructurados y semi-estructurados (por ejemplo json).
Sistema NoSQL	Pertenece a las bases de datos NoSQL, ideal para grandes volúmenes de datos y escalabilidad horizontal.
Lenguaje de Consulta (CQL)	Usa Cassandra Query Language (CQL), similar a SQL, diseñado para aprovechar su modelo basado en columnas.
Alta Disponibilidad	Redundancia integrada con replicación automática en múltiples nodos, garantizando acceso continuo a los datos. (Es decir lo que se logra con esto es que los datos siempre estén disponibles incluso con la falla de algún nodo)
Distribuido y Descentralizado	Todos los nodos en el clúster tienen el mismo rol y capacidades ("peer-to-peer"). Esto elimina la necesidad de un nodo maestro central que controle el sistema, como ocurre en otras arquitecturas (masterless).
Escalabilidad Horizontal	Los nodos pueden añadirse al clúster sin interrupción, incrementando la capacidad de forma lineal.
Consistencia Tunable	Permite ajustar el nivel de consistencia según las necesidades, desde consistencia eventual hasta estricta. (Según la aplicación se ajustan de forma adecuada las medidas de rendimiento, disponibilidad ...)

Optimización para Escrituras	Diseñada para manejar grandes volúmenes de escrituras con alta velocidad, gracias al uso de commit logs y SSTables .
Tolerancia a Fallos	El sistema continúa funcionando incluso si varios nodos fallan, gracias a la replicación de datos.
Soporte Multi Datacenter	Compatible con replicación en múltiples datacenters para alta disponibilidad geográfica.
Modelo BASE	Sigue el enfoque BASE (Basically Available, Soft state, Eventually consistent) en lugar de ACID.
Código Abierto	Es un proyecto de código abierto bajo la licencia Apache, con una amplia comunidad de soporte.
Usos Típicos	Redes sociales, IoT, comercio electrónico, análisis en tiempo real, almacenamiento

CONCEPTOS ACERCA DE CASSANDRA

- **Modelo Base:** es el enfoque que sigue cassandra a diferencia del modelo ACID que es el tradicional de las bases de datos tradicionales, este modelo **prioriza la alta disponibilidad y el rendimiento en sistemas distribuidos**, aceptando que los datos pueden no estar perfectamente sincronizados entre todos los nodos en todo momento (**soft state**), pero eventualmente alcanzarán la consistencia (**eventual consistency**).
- **Commit logs:** son archivos de registro que almacenan todas las operaciones de escritura que se realizan en una base de datos antes de que los datos sean

definitivamente escritos en disco. Esto garantiza **durabilidad** haciendo que no se pierdan los datos en caso de fallo del sistema.

- **SSTables:** son archivos **inmutables** en los que se almacenan de manera permanente los datos una vez que han sido procesados y organizados. Cuando los datos se escriben en Cassandra, primero se almacenan en memoria en una estructura llamada **MemTable**. Cuando esta MemTable alcanza su capacidad límite, los datos se vuelcan a disco en forma de una SSTable.
- **Masterless:** se refiere a una arquitectura en la que no existe un nodo central o principal que tenga control exclusivo sobre la base de datos o el sistema. En un sistema **masterless**, todos los nodos son iguales y tienen la misma responsabilidad, lo que significa que no hay un único punto de falla.
- **Peer to peer:** arquitectura en la que todos los nodos en el clúster son iguales y se comunican directamente entre sí, sin la necesidad de un nodo central o maestro. (Esta tecnología permite que los nodos cooperen para distribuir y replicar los datos, lo que garantiza que no haya un único punto de falla y mejore la disponibilidad y escalabilidad del sistema)

¿PARA QUÉ SE USA CASSANDRA?

Cassandra se utiliza en la actualidad **principalmente en aplicaciones que requieren manejar grandes volúmenes de datos distribuidos a escala, con alta disponibilidad y tolerancia a fallos**. Es ideal para entornos donde los datos están distribuidos geográficamente y deben estar siempre disponibles, como en **plataformas de redes sociales, aplicaciones móviles, servicios financieros, análisis en tiempo real y comercio electrónico**.

¿QUÉ EMPRESAS LA USAN EN EL MERCADO?

Algunas de las empresas más destacadas que utilizan Cassandra incluyen **Netflix**, para gestionar sus catálogos de contenido y recomendaciones de usuarios; **Instagram**, que la emplea para manejar las interacciones y los datos de usuario; **eBay**, para gestionar las transacciones en tiempo real; **Uber**, para manejar las solicitudes de viajes y datos de ubicación en tiempo real; y **Spotify**, para almacenar y procesar datos de usuario y preferencias musicales.

¿QUÉ USO REAL SE LE PODRÍA DAR?

Se podría usar en **análisis en tiempo real** para procesar datos generados de forma continua, como logs de servidores o eventos de usuarios. También es ideal para **sistemas de recomendación** en plataformas de comercio electrónico o contenido multimedia, donde se puede gestionar eficientemente datos de usuarios y sus preferencias. Además, podría usarse para **gestionar grandes volúmenes de datos de clientes** en entornos de marketing, ventas o servicios financieros. También es excelente para **sistemas de monitoreo y análisis de logs**, garantizando alta disponibilidad y rendimiento incluso en infraestructuras complejas.

INSTALACIÓN Y ADMINISTRACIÓN

PASOS A SEGUIR PARA INSTALARLO

1º. Actualizamos los paquetes

```
root@Tifa:~# sudo apt update
Get:1 http://mirror.hetzner.com/debian/packages bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [151 kB]
```

2º. Añadimos el siguiente repositorio a **/etc/apt/sources.list** y actualizamos los paquetes.

```
GNU nano 7.2                                         /etc/apt/sources.list *
deb http://deb.debian.org/debian bookworm main contrib non-free-firmware
# deb-src http://deb.debian.org/debian bookworm main contrib non-free-firmware

deb http://deb.debian.org/debian bookworm-updates main contrib non-free-firmware
# deb-src http://deb.debian.org/debian bookworm-updates main contrib non-free-firmware

# deb http://deb.debian.org/debian bookworm-backports main contrib non-free-firmware
# deb-src http://deb.debian.org/debian bookworm-backports main contrib non-free-firmware

deb http://security.debian.org/debian-security bookworm-security main contrib non-free-firmware
# deb-src http://security.debian.org/debian-security bookworm-security main contrib non-free-firmware
deb http://deb.debian.org/debian unstable main non-free contrib # Añadimos esto
```

```
root@Tifa:~# sudo apt update
Hit:1 http://mirror.hetzner.com/debian/packages bookworm InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
```

3º. Instalaremos Open JDK 11 package debido a que es un paquete necesario para que funcione Cassandra (Cassandra necesita de **Java** para poder funcionar es debido a eso).

```
root@Tifa:~# sudo apt install openjdk-11-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
base-files base-passwd bsdxtrautils bsduils ca-certificates-java eject
fdisk fontconfig-config fonts-dejavu-core fonts-dejavu-mono gcc-14-base
java-common libasound2-data libasound2t64 libatomic1 libavahi-client3
libavahi-common-data libavahi-common3 libblkid1 libc-bin libc-l10n libc6
libcups2t64 libdrm-amdpu1 libdrm-common libdrm-intel1 libdrm-radeon1
libdrm2 libelf1t64 libfdisk1 libfontconfig1 libgbm1 libgcc-s1 libgif7
libgl1 libgl1-mesa-dri libglapi-mesa libglib2.0-0t64 libglib2.0-data
libglvnd0 libglx-mesa0 libglx0 libgmp10 libgnutls30t64 libgraphite2-3
libharfbuzz0b libhogweed6t64 libjpeg62-turbo liblcms2-2 libllvm19 libmount1
libnettle8t64 libnpn4 libncursesw-systemd libnss3 libnss3-modified libnss3-systemd
```

Y comprobamos la versión para ver si se ha instalado correctamente.

```
root@Tifa:~# java --version
openjdk 11.0.26-ea 2025-01-21
OpenJDK Runtime Environment (build 11.0.26-ea+6-post-Debian-1)
OpenJDK 64-Bit Server VM (build 11.0.26-ea+6-post-Debian-1, mixed mode, sharing )
root@Tifa:~#
```

4º. Añadimos el repositorio de cassandra a **/etc/apt/sources.list**

```
root@Tifa:~# echo "deb [signed-by=/etc/apt/keyrings/apache-cassandra.asc] https://debian.cassandra.apache.org 41x main" | sudo tee -a /etc/apt/sources.list.d/cassandra.sources.list
deb [signed-by=/etc/apt/keyrings/apache-cassandra.asc] https://debian.cassandra.apache.org 41x main
root@Tifa:~#
```

5º. Descargamos la clave de la firma del repositorio de Cassandra

```
root@Tifa:~# sudo curl -o /etc/apt/keyrings/apache-cassandra.asc https://downloads.apache.org/cassandra/KEYS
% Total    % Received % Xferd  Average Speed   Time     Time      Time Current
          Dload  Upload Total   Spent    Left  Speed
100  256k  100  256k    0      0  3731k      0 --:--:-- --:--:-- --:--:-- 3821k
root@Tifa:~#
```

```
root@Tifa:~# sudo apt update
Hit:1 http://mirror.hetzner.com/debian/packages bookworm InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Hit:3 http://security.debian.org/debian-security bookworm-security InRelease
Hit:4 http://mirror.hetzner.com/debian/packages bookworm-updates InRelease
Hit:5 http://mirror.hetzner.com/debian/security bookworm-security InRelease
Hit:6 http://deb.debian.org/debian bookworm-updates InRelease
Hit:7 http://deb.debian.org/debian unstable InRelease
```

6º. Instalamos cassandra

```
root@Tifa:~# sudo apt install cassandra
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  cassandra-tools
The following NEW packages will be installed:
  cassandra
```

Y comprobamos que se ha iniciado con éxito

```
root@Tifa:~# sudo systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
   Loaded: loaded (/etc/init.d/cassandra; generated)
   Active: active (running) since Wed 2025-01-08 17:31:30 UTC; 16s ago
     Invocation: e4b834c4ac924fbf9be1aab10f372c34
       Docs: man:systemd-sysv-generator(8)
    Process: 20058 ExecStart=/etc/init.d/cassandra start (code=exited, status=>
      Tasks: 44 (limit: 4531)
     Memory: 1.2G (peak: 1.2G)
        CPU: 21.603s
      CGroup: /system.slice/cassandra.service
              └─20162 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriorit...
```

Jan 08 17:31:29 Tifa systemd[1]: Starting cassandra.service - LSB: distributed...
Jan 08 17:31:30 Tifa systemd[1]: Started cassandra.service - LSB: distributed...
[lines 1-14/14 (END)]

Además comprobaremos el estado del nodo del clúster cassandra para ver si está activo.

(Es uno de los pasos más importantes porque si no está activo puede ser por errores)

```
root@Tifa:~# sudo nodetool status
Datacenter: datacenter1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address      Load      Tokens  Owns (effective)  Host ID          Rack
UN  127.0.0.1    104.37  KiB    16        100.0%          4b602aae-8067-4570-b4aa-ca4546d08378  rack1
root@Tifa:~#
```

PARADA Y ARRANQUE DE CASSANDRA

1º. Comprobaremos el estado de cassandra (**systemctl status cassandra**)

```
root@Tifa:~# sudo systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
  Loaded: loaded (/etc/init.d/cassandra; generated)
  Active: active (running) since Wed 2025-01-08 17:31:30 UTC; 5min ago
    Invocation: e4b834c4ac924fbf9be1aab10f372c34
      Docs: man:systemd-sysv-generator(8)
    Process: 20058 ExecStart=/etc/init.d/cassandra start (code=exited, status=>
      Tasks: 50 (limit: 4531)
     Memory: 1.3G (peak: 1.3G)
       CPU: 38.861s
      CGroup: /system.slice/cassandra.service
              └─20162 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriori>

Jan 08 17:31:29 Tifa systemd[1]: Starting cassandra.service - LSB: distributed>
Jan 08 17:31:30 Tifa systemd[1]: Started cassandra.service - LSB: distributed >
lines 1-14/14 (END)
```

2º. Pararemos cassandra (**systemctl stop cassandra**)

```
root@Tifa:~# sudo systemctl stop cassandra
root@Tifa:~# sudo systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
  Loaded: loaded (/etc/init.d/cassandra; generated)
  Active: inactive (dead) since Wed 2025-01-08 17:36:58 UTC; 2s ago
    Duration: 5min 23.449s
  Invocation: e4b834c4ac924fbf9be1aab10f372c34
    Docs: man:systemd-sysv-generator(8)
  Process: 20058 ExecStart=/etc/init.d/cassandra start (code=exited, status=>
  Process: 20850 ExecStop=/etc/init.d/cassandra stop (code=exited, status=0/>
  Mem peak: 1.3G
    CPU: 40.095s

Jan 08 17:31:29 Tifa systemd[1]: Starting cassandra.service - LSB: distributed>
Jan 08 17:31:30 Tifa systemd[1]: Started cassandra.service - LSB: distributed >
Jan 08 17:36:53 Tifa systemd[1]: Stopping cassandra.service - LSB: distributed>
Jan 08 17:36:58 Tifa systemd[1]: cassandra.service: Deactivated successfully.
Jan 08 17:36:58 Tifa systemd[1]: Stopped cassandra.service - LSB: distributed >
Jan 08 17:36:58 Tifa systemd[1]: cassandra.service: Consumed 40.095s CPU time,>
lines 1-17/17 (END)
```

3º. La iniciaremos (`systemctl start cassandra`)

```
root@Tifa:~# sudo systemctl start cassandra
root@Tifa:~# sudo systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
    Loaded: loaded (/etc/init.d/cassandra; generated)
      Active: active (running) since Wed 2025-01-08 17:37:58 UTC; 2s ago
        Invocation: eab7e575d1a544caa9d334a302ee52d0
          Docs: man:systemd-sysv-generator(8)
      Process: 20883 ExecStart=/etc/init.d/cassandra start (code=exited, status=0)
        Tasks: 16 (limit: 4531)
       Memory: 1.1G (peak: 1.1G)
         CPU: 3.382s
      CGroup: /system.slice/cassandra.service
              └─20982 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriorities
Jan 08 17:37:58 Tifa systemd[1]: Starting cassandra.service - LSB: distributed...
Jan 08 17:37:58 Tifa systemd[1]: Started cassandra.service - LSB: distributed...
[lines 1-14/14 (END)]
```

UBICACIÓN DE ARCHIVOS GESTIÓN POR DEFECTO

Los datos de Apache Cassandra se almacenan en el `/var/lib/cassandra` como se puede ver en la imagen.

```
root@Tifa:~# ls /var/lib/cassandra/
commitlog  data  hints  saved_caches
root@Tifa:~#
```

- **commitlog:** Contiene los logs de escritura de Cassandra para garantizar durabilidad. Los datos se escriben aquí antes de confirmarse en los nodos.
- **data:** Almacena los datos estructurados y persistidos en tablas, organizados por keyspace y tabla.
- **hints:** Guarda pistas para replicación diferida, permitiendo que nodos caídos reciban datos pendientes al volver en línea.
- **saved_caches:** Contiene cachés guardadas para mejorar el rendimiento en consultas frecuentes, como índices o particiones.

Los archivos de configuración se encuentran en **/etc/cassandra**

```
root@Tifa:~# ls /etc/cassandra/
cassandra-env.sh                      jvm11-server.options
cassandra-rackdc.properties            jvm8-clients.options
cassandra-topology.properties.example  jvm8-server.options
cassandra.yaml                         jvm-clients.options
commitlog_archiving.properties         jvm-server.options
cqlshrc.sample                         logback-tools.xml
credentials.sample                     logback.xml
hotspot_compiler                       triggers
jvm11-clients.options
root@Tifa:~#
```

- **cassandra-env.sh:** Configura variables de entorno de Cassandra, como opciones JVM y rutas de archivos.
- **cassandra-rackdc.properties:** Define el rack y datacenter del nodo para la estrategia de replicación.
- **cassandra-topology.properties.example:** Ejemplo de configuración de topología para mapear nodos a racks y datacenters.
- **cassandra.yaml:** Archivo principal de configuración que define parámetros clave, como almacenamiento, nodos, y estrategias de replicación.
- **commitlog_archiving.properties:** Configura políticas para archivar y recuperar los commit logs.
- **cqlshrc.sample:** Archivo de ejemplo para configurar preferencias del cliente **cqlsh**, como autenticación y formato de salida.
- **credentials.sample:** Plantilla para configurar credenciales predeterminadas de usuarios y roles.
- **hotspot_compiler:** Ajusta configuraciones del compilador HotSpot de la JVM (**Java virtual machine**) para optimización de rendimiento.
- **jvm11-clients.options:** Configura opciones JVM específicas para clientes ejecutándose con Java 11.
- **jvm11-server.options:** Configura opciones JVM específicas para el servidor ejecutándose con Java 11.

- **jvm8-clients.options:** Configura opciones JVM específicas para clientes ejecutándose con Java 8.
- **jvm8-server.options:** Configura opciones JVM específicas para el servidor ejecutándose con Java 8.
- **jvm-clients.options:** Opciones JVM genéricas aplicables a clientes, independientemente de la versión de Java.
- **jvm-server.options:** Opciones JVM genéricas aplicables al servidor, independientemente de la versión de Java.
- **logback-tools.xml:** Configura el sistema de registro para herramientas relacionadas con Cassandra.
- **logback.xml:** Configura el sistema de registro principal de Cassandra, definiendo niveles y destinos de logs.
- **triggers:** Directorio que contiene scripts o clases personalizadas para ejecutar triggers en las operaciones de datos.

Las opciones de inicio de Java se pueden configurar en el **/etc/default/cassandra**; también se pueden configurar en **/etc/cassandra/cassandra-env.sh**

```
GNU nano 7.2          /etc/default/cassandra
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#
# NOTICE: See also /etc/cassandra/cassandra-env.sh
#
# EXTRA_CLASSPATH provides the means to extend Cassandra's classpath with
# additional libraries. It is formatted as a colon-delimited list of
# class directories and/or jar files. For example, to enable the
# JMX-to-web bridge install libmx4j-java and uncomment the following.
#EXTRA_CLASSPATH="/usr/share/java/mx4j-tools.jar"
```

Dentro de este fichero se pueden usar una serie de parámetros:

- **MAX_HEAP_SIZE**: Define el tamaño máximo del heap de Java, crucial para gestionar la memoria asignada a Cassandra.
- **JVM_OPTS**: Permite personalizar varias opciones JVM, como habilitar recolección de basura avanzada o configuraciones de afinidad de CPU.
- **DATA_DIRS**: Configura rutas personalizadas para los directorios de datos de Cassandra. Útil si necesitas separar cargas de escritura/lectura.
- **GC_LOGGING_OPTS**: Habilita y configura el registro de la recolección de basura para análisis de rendimiento
- **JVM_EXTRA_OPTS**: Agrega configuraciones adicionales específicas para entornos particulares, como límites de threads o mejoras de seguridad.

CONEXIONES, ALMACENAMIENTO, ENTORNO, VARIABLES, PUERTOS...

Conexiones

Utiliza el lenguaje **CQL (Cassandra Query Language)**, similar a SQL, para interactuar con los datos. Desde su consola, llamada **cqlsh**, los usuarios pueden ejecutar comandos para crear esquemas, insertar, consultar y gestionar datos. Esta consola se conecta a un nodo del clúster y permite realizar operaciones de lectura o escritura de manera directa. Cassandra distribuye los datos entre nodos utilizando un **modelo de particionado** basado en claves y replica la información según la estrategia configurada, garantizando tolerancia a fallos.

Contando con una gran variedad de conexiones en **Cassandra** haré como ejemplo la creación de una bd y una tabla y probaré conexiones a esta base de datos de forma remota tanto de manera simple como mediante un script en python.

1º. Abriremos la consola de **cassandra**

```
root@Tifa:~# cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh>
```

2º. Crearemos una “**base de datos**” que es **Cassandra** es llamada **Keyspace**

my_keyspace: Nombre del keyspace/bd.

SimpleStrategy: Estrategia de replicación (apta para entornos de un solo nodo).

replication_factor=1: Solo una réplica por nodo.

```
cqlsh> CREATE KEYSPACE my_keyspace  
... WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};  
cqlsh>
```

3º. Usamos el **keyspace**

```
cqlsh> USE my_keyspace;  
cqlsh:my_keyspace>
```

4º. Una vez dentro de del **keyspace** crearemos una **tabla**

```
cqlsh:my_keyspace> CREATE TABLE users (  
...         id UUID PRIMARY KEY,  
...         name TEXT,  
...         email TEXT,  
...         age INT  
... );  
cqlsh:my_keyspace>
```

Comprobaremos que se ha creado la tabla

```
cqlsh:my_keyspace> DESCRIBE TABLES;  
  
users  
  
cqlsh:my_keyspace>
```

Y también comprobaremos su contenido

```
cqlsh:my_keyspace> DESCRIBE TABLE users;

CREATE TABLE my_keyspace.users (
    id uuid PRIMARY KEY,
    age int,
    email text,
    name text
) WITH additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCo
mpactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra
.io.compress.LZ4Compressor'}
    AND memtable = 'default'
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
cqlsh:my_keyspace> █
```

5º. Insertamos datos en esta tabla

```
cqlsh:my_keyspace> INSERT INTO users (id, name, email, age) VALUES (uuid(), 'Ni  
colás Álvarez', 'nicolas@example.com', 19);  
cqlsh:my keyspace> █
```

Y comprobamos su contenido

```
cqlsh:my_keyspace> SELECT * FROM users;
 id | age | email | name
---+---+---+---
 3a2d1000-5cd1-4d35-941a-d3e9e1b22d39 | 19 | nicolas@example.com | Nicolás Álvarez
(1 rows)
cqlsh:my_keyspace>
```

6º. Una vez creada el **keyspace**, configuraremos a **Cassandra** para que esta nos permita conexiones externas como puede ser un **script python**

```
root@Tifa:~# nano /etc/cassandra/cassandra.yaml
```

```
#  
# For security reasons, you should not expose this  
rpc_address: 138.199.149.188
```

rpc_address: 138.199.149.188 → Especifica la dirección IP en la que el nodo escucha conexiones de cliente para operaciones CQL. En este caso, **138.199.149.188** indica que las conexiones remotas usarán esta IP para interactuar con el nodo.

```
root@Tifa:~# systemctl restart cassandra
root@Tifa:~# systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
    Loaded: loaded (/etc/init.d/cassandra; generated)
    Active: active (running) since Thu 2025-01-09 08:44:28 UTC; 14s ago
      Invocation: e3f0b93a914846129e209326d2e3d898
        Docs: man:systemd-sysv-generator(8)
   Process: 25703 ExecStart=/etc/init.d/cassandra start (code=exited, status=0/SUCCESS)
     Tasks: 48 (limit: 4531)
    Memory: 1.3G (peak: 1.3G)
      CPU: 23.202s
     CGroup: /system.slice/cassandra.service
             └─25803 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriorities -XX:-
Jan 09 08:44:28 Tifa systemd[1]: Starting cassandra.service - LSB: distributed storage >
Jan 09 08:44:28 Tifa systemd[1]: Started cassandra.service - LSB: distributed storage >
lines 1-14/14 (END)
```

7º. Nos conectaremos en un primer momento desde la misma máquina para verificar la conectividad.

```
(None, None) connecting to [::1:138.199.149.188], 9042]]). Last error: Connection
root@Tifa:~# cqlsh 138.199.149.188
Connected to Test Cluster at 138.199.149.188:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh>
```

8º. Además de comprobar que se puede acceder desde el mismo vps probaré desde otro, para ello crearé un entorno virtual

```
root@EzioCloud:~# mkdir ~mi_proyecto
cd ~mi_proyecto
root@EzioCloud:~/mi_proyecto# python3 -m venv mi_entorno
root@EzioCloud:~/mi_proyecto# source mi_entorno/bin/activate
(mi_entorno) root@EzioCloud:~/mi_proyecto#
```

Instalaremos el siguiente paquete para poder conectarme a la consola de **Cassandra** de forma remota.

```
root@EzioCloud:~/mi_proyecto# source mi_entorno/bin/activate
(mi_entorno) root@EzioCloud:~/mi_proyecto# pip install cqlsh
Collecting cqlsh
  Downloading cqlsh-6.2.0-py3-none-any.whl (105 kB)
                                             105.5/105.5 kB 3.9 MB/s eta 0:00:00
Collecting cassandra-driver
  Downloading cassandra_driver-3.29.2-cp311-cp311-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (3.9 MB)
                                             3.9/3.9 kB 20.3 MB/s eta 0:00:00
Collecting pure-sasl
```

Y probaré la conectividad

```
(mi_entorno) root@EzioCloud:~/mi_proyecto# cqlsh 138.199.149.188
WARNING: cqlsh was built against 5.0.0, but this server is 4.1.7. All features
may not work!
Connected to Test Cluster at 138.199.149.188:9042
[cqlsh 6.2.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> use my_keyspace;
cqlsh:my_keyspace> describe tables;

users

cqlsh:my_keyspace>
```

También probaré a conectarme con un script en python

```
(mi_entorno) root@EzioCloud:~/mi_proyecto# pip install cassandra-driver
Requirement already satisfied: cassandra-driver in ./mi_entorno/lib/python3.11/
site-packages (3.29.2)
Requirement already satisfied: geomet<0.3,>=0.1 in ./mi_entorno/lib/python3.11/
site-packages (from cassandra-driver) (0.2.1.post1)
Requirement already satisfied: click in ./mi_entorno/lib/python3.11/site-packages
```



```
GNU nano 7.2                                         conexioncassandra.py
from cassandra.cluster import Cluster

# Configuración de conexión
CASSANDRA_IP = "138.199.149.188" # IP del servidor Cassandra
CASSANDRA_PORT = 9042 # Puerto por defecto de Cassandra

def connect_and_list_keyspaces():
    try:
        # Conexión al clúster sin autenticación
        cluster = Cluster([CASSANDRA_IP], port=CASSANDRA_PORT)
        session = cluster.connect()

        # Mostrar la versión de Cassandra
        rows = session.execute('SELECT release_version FROM system.local')
        for row in rows:
            print(f"Cassandra version: {row.release_version}")

        print("\n--- Keyspaces en Cassandra ---")

        # Listar los keyspaces disponibles
        keyspaces = session.execute('SELECT keyspace_name FROM system_schema.keyspaces')
        for keyspace in keyspaces:
            print(f"- {keyspace.keyspace_name}")

        # Cerrar conexión
        cluster.shutdown()

    except Exception as e:
        print(f"Error al conectarse a Cassandra: {e}")

if __name__ == "__main__":
    connect_and_list_keyspaces()
```

Y al ejecutarlo me proporciona la versión y las bases de datos que tiene/ keyspaces (me proporciona eso porque creo que es una manera en la que se puede verificar que la conexión funciona de forma adecuada)

```
(mi_entorno) root@EzioCloud:~/mi_proyecto# python3 conexioncassandra.py
Cassandra version: 4.1.7

--- Keyspaces en Cassandra ---
- system_auth
- system_schema
- system_distributed
- my_keyspace
- system
- system_traces
(mi_entorno) root@EzioCloud:~/mi_proyecto#
```

Almacenamiento

Cassandra utiliza un almacenamiento basado en una **arquitectura de tablas de columnas, en lugar de filas**, lo que **mejora la eficiencia en las consultas de grandes volúmenes** de datos. Los **datos se dividen en particiones**, donde cada partición es determinada por la clave de partición. Dentro de cada partición, las filas se ordenan mediante una clave de clustering. Los datos se almacenan en **MemTables** en memoria, que se escriben a disco como **SSTables** una vez que se llenan. Cassandra utiliza commit logs para garantizar la durabilidad de los datos y ejecuta un proceso de **compaction** para consolidar las **SSTables**, eliminar datos obsoletos y optimizar el uso del espacio en disco. Además, el sistema maneja la replicación de datos entre nodos para asegurar alta disponibilidad y tolerancia a fallos.

Para poder ver los distintos tipos de almacenamiento en cassandra se pueden usar los siguientes comandos:

Keyspaces (bases de datos)

```
cqlsh> DESCRIBE KEYSPACES;

my_keyspace  system_auth      system_schema  system_views
system       system_distributed system_traces  system_virtual_schema
```

Ver tablas dentro de un keyspace

```
cqlsh> use my_keyspace;
cqlsh:my_keyspace> describe tables;

users
```

Ver la estructura de la tabla

```
cqlsh:my_keyspace> describe table users;

CREATE TABLE my_keyspace.users (
    id uuid PRIMARY KEY,
    age int,
    email text,
    name text
) WITH additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCo
mpactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassand
ra.io.compress.LZ4Compressor'}
    AND memtable = 'default'
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
cqlsh:my_keyspace>
```

Inspeccionar los datos

```
cqlsh:my_keyspace> SELECT * FROM users LIMIT 10;

id                               | age | email                | name
-----+-----+-----+-----+
3a2d1000-5cd1-4d35-941a-d3e9e1b22d39 | 19 | nicolas@example.com | Nicolás Álvarez
(1 rows)
```

Estado del cluster, carga del nodo y distribución de los datos

```
root@Tifa:~# nodetool status
Datacenter: datacenter1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address   Load   Tokens  Owns (effective)  Host ID
UN  127.0.0.1  161.44 KiB    16      100.0%          4b602aae-8067-4570-b4aa-ca4546d08378  rack1
root@Tifa:~#
```

Información detallada sobre la distribución de las particiones en el anillo

(Cassandra utiliza un **algoritmo de particionado** que asigna una **clave de partición** a cada **registro**. Estas **claves** son mapeadas a un **rango de tokens**, y **cada nodo** del clúster es **responsable de un rango específico** dentro de un espacio de clave virtual, **conocido como el anillo.**)

```
root@Tifa:~# nodetool ring
Datacenter: datacenter1
=====
Address   Rack   Status State   Load       Owns            Token
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  9190701150880730742
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -8176670115381219215
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -6933150109700627241
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -5965785866622091742
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -4375781314108193104
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -2756758018411149807
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  -1778612565895035732
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  21734928551396
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  1096063635163489685
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  2187360745855701944
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  3031737393346863701
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  4472005623249446939
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  5543506239589479752
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  6561357644331906767
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  7379029576944257562
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  8454785458169837101
127.0.0.1  rack1  Up     Normal  161.44 KiB  100.00%  9190701150880730742
```

Ver estadísticas detalladas sobre una columna específica (column family) de una tabla

```
root@Tifa:~# nodetool cfstats my_keyspace.users
Total number of tables: 45
-----
Keyspace : my_keyspace
  Read Count: 0
  Read Latency: NaN ms
  Write Count: 0
  Write Latency: NaN ms
  Pending Flushes: 0
    Table: users
      SSTable count: 1
      Old SSTable count: 0
      Space used (live): 5169
      Space used (total): 5169
      Space used by snapshots (total): 0
      Off heap memory used (total): 44
      SSTable Compression Ratio: 1.0875
      Number of partitions (estimate): 1
      Memtable cell count: 0
      Memtable data size: 0
      Memtable off heap memory used: 0
      Memtable switch count: 0
      Speculative retries: 0
      Local read count: 0
      Local read latency: NaN ms
      Local write count: 0
      Local write latency: NaN ms
      Pending flushes: 0
      Percent repaired: 0.0
      Bytes repaired: 0.000KiB
      Bytes unrepairs: 0.078KiB
      Bytes pending repair: 0.000KiB
      Bloom filter false positives: 0
      Bloom filter false ratio: 0.00000
      Bloom filter space used: 16
      Bloom filter off heap memory used: 8
      Index summary off heap memory used: 28
      Compression metadata off heap memory used: 8
      Compacted partition minimum bytes: 73
      Compacted partition maximum bytes: 86
      Compacted partition mean bytes: 86
      Average live cells per slice (last five minutes): NaN
      Maximum live cells per slice (last five minutes): 0
      Average tombstones per slice (last five minutes): NaN
      Maximum tombstones per slice (last five minutes): 0
      Dropped Mutations: 0
      Droppable tombstone ratio: 0.00000
-----
root@Tifa:~#
```

También es importante destacar que se puede monitorizar con **Herramientas de Monitoreo**:

Se puede usar herramientas como **Cassandra Management Console**, **DataStax OpsCenter**, **Prometheus**, o **Grafana**. (Entre bastantes más herramientas)

Entornos y variables

En un primer momento mostraré la información sobre otros nodos del clúster (en mi caso no tengo ningún nodo más unido en este momento) **select * from system.peers**.

```
(0 rows)
admin_user@cqlsh> SELECT * FROM system.peers;
  peer | data_center | host_id | preferred_ip | rack | release_version | rpc_address | schema_version | tokens
-----+-----+-----+-----+-----+-----+-----+-----+-----+
(0 rows)
admin_user@cqlsh> A
```

- **peer**: IP del nodo remoto en el clúster.
- **data_center**: Datacenter al que pertenece el nodo.
- **host_id**: Identificador único del nodo en el clúster.

- **preferred_ip**: IP preferida para comunicación entre nodos.
 - **rack**: Rack dentro del datacenter donde está el nodo.
 - **release_version**: Versión de Cassandra que ejecuta el nodo.
 - **rpc_address**: IP para solicitudes RPC (cliente-servidor).

Ahora mostraré la información acerca del nodo local (el nodo desde donde ejecutas la consulta). Este nodo almacena datos relacionados con su configuración, estado y roles dentro del clúster.

- **key:** Representa la clave "local", que indica que los datos corresponden al nodo actual.
 - **bootstrapped:** Indica si el nodo ha completado el proceso de unión al clúster (COMPLETED significa que está operativo).
 - **broadcast_address:** Dirección IP que el nodo anuncia al resto del clúster para comunicarse.
 - **broadcast_port:** Puerto usado por el nodo para la comunicación inter-nodos (normalmente 7000).
 - **cluster_name:** Nombre del clúster al que pertenece el nodo.
 - **cql_version:** Versión del lenguaje CQL que soporta el nodo (por ejemplo, 3.4.6).
 - **data_center:** Datacenter al que pertenece el nodo, útil en configuraciones multi-datacenter.
 - **gossip_generation:** Valor que indica el tiempo (en segundos desde la época Unix) en que el nodo inició el protocolo Gossip.

- **host_id:** Identificador único del nodo (UUID), independiente de la dirección IP.
- **listen_address:** Dirección IP en la que el nodo escucha conexiones internas (de otros nodos).
- **listen_port:** Puerto usado por el nodo para escuchar conexiones internas (normalmente 7000).
- **native_protocol_version:** Versión del protocolo nativo de Cassandra usado para las comunicaciones cliente-servidor.
- **partitioner:** Tipo de particionador usado para distribuir datos entre los nodos (por defecto Murmur3Partitioner).
- **rack:** Rack al que pertenece el nodo dentro del datacenter, usado para replicación y tolerancia a fallos.
- **release_version:** Versión del software Cassandra que ejecuta el nodo.
- **rpc_address:** Dirección IP para las conexiones RPC (usada por clientes para interactuar con Cassandra).
- **rpc_port:** Puerto para conexiones RPC (normalmente 9042).
- **schema_version:** Identificador de la versión del esquema del clúster; cambia cuando se modifica el esquema.
- **tokens:** Lista de tokens asignados al nodo, que determinan qué datos almacena en el clúster.

Por ejemplo para mostrar los tokens (**select tokens from system.local**)

```
admin_user@cqlsh> SELECT tokens FROM system.local;
tokens
-----
(-1778612565895035732, -2756758018411149887, -4375781314108193104, -5965785866622891742, -6933150189708627241, -8176670115381219215, -1096663635163489685, '21734928551396', '2187360745855701944', '3031739
346863781, 447205623249446939, 5543560239589479752, 6561357644331986767, 7379829576944257962, 8454785458169837181, 919078156888730742)
(1 rows)
admin_user@cqlsh>
```

Puertos

Cassandra utiliza/utilizaba varios puertos que son esenciales para sus operaciones:

- **Puerto 7000 (interno):** Se usa para la comunicación entre nodos en un clúster (protocolo Gossip). Si la comunicación está cifrada (SSL), se utiliza el puerto **7001** en su lugar.
- **Puerto 9042 (RPC):** Es el puerto principal para conexiones cliente-servidor mediante el protocolo CQL (Cassandra Query Language).

- **Puerto 7199 (JMX)**: Utilizado para la gestión y monitorización de Cassandra mediante Java Management Extensions (JMX).
- **Puerto 9160 (Thrift)**: Puerto antiguo para comunicación cliente-servidor usando el protocolo Thrift (descontinuado en versiones recientes).
- **Puerto 9142 (alternativo para RPC)**: En configuraciones personalizadas, puede usarse como una alternativa al puerto 9042 para conexiones CQL.

Una vez sabemos los puertos, ahora lo necesario es saber donde ponerlos para poder usarlos si tenemos la necesidad, para ello hay que editar **/etc/cassandra/cassandra.yaml** (menos el puerto **JMX**, suelen venir por defecto pero he puesto un ejemplo de cómo sería)

```
GNU nano 7.2                                         /etc/cassandra/cassandra.yaml *
-----Puertos Cassandra -----
# Puerto Gossip
storage_port: 7000
ssl_storage_port: 7001

# Puerto para conexiones RCP (cliente-servidor)
rpc_port: 9042

# Puerto JMX (Gestión y monitorización, este se configura en /etc/cassandra/cassandra-env.sh)
JVM_OPTS="$JVM_OPTS -Dcom.sun.management.jmxremote.port=7199"

# Puerto Thrift (necesario tener start_rpc:true)
thrift_port: 9160
```

Nomenclatura de administración desde cli del sistema

La nomenclatura que pueden ser comandos o conceptos más importantes la mostraré en la siguiente tabla, donde la resumiré lo mejor que pueda los comandos principales.

COMANDOS	DESCRIPCIÓN
CREATE KEYSPACE	Crea un keyspace, que es un contenedor para tablas, con estrategias de replicación configuradas.
USE	Cambia al keyspace especificado para ejecutar comandos en él.
CREATE TABLE	Crea una tabla dentro del keyspace, definiendo columnas y claves primarias.
INSERT	Inserta una fila en una tabla con valores específicos para cada columna.
SELECT	Recupera datos de una tabla, permitiendo filtros y proyecciones de columnas.

UPDATE	Actualiza los valores de columnas en una fila específica.
DELETE	Elimina una fila o columnas específicas de una tabla.
DROP	Elimina un keyspace o una tabla de forma permanente.
TRUNCATE	Elimina todos los datos de una tabla, pero mantiene su estructura.
ALTER TABLE	Modifica la estructura de una tabla, como agregar o eliminar columnas.
CREATE INDEX	Crea un índice en una columna de una tabla para mejorar la eficiencia de las consultas.
DESCRIBE	Muestra la estructura de un keyspace, tabla u objeto en Cassandra.
CONSISTENCY	Establece el nivel de consistencia para las consultas en la sesión actual.
LIST USERS	Muestra los usuarios y roles configurados en Cassandra.
CREATE ROLE / GRANT	Crea roles de usuario y asigna permisos específicos en un keyspace o tabla.
nodetool status	Muestra el estado de los nodos en el clúster, incluyendo su carga, estado de tokens y conectividad.
nodetool repair	Sincroniza los datos replicados en nodos para mantener la consistencia.
nodetool flush	Fuerza la escritura de datos en memoria (memtable) al disco (SSTable).
nodetool cleanup	Limpia los datos obsoletos en nodos después de una operación de rediseño del clúster.

HERRAMIENTA GRÁFICA Y EXPLICACIÓN DE LA ARQUITECTURA USADA

Antes de explicar lo siguiente, he tenido que pasarme a máquinas virtuales ya que los recursos de lo que quería montar que es una demo que trás investigar la encontré son bastante superiores de los que dispone mi VPS, además cabe recalcar que he usado un Clúster de prueba que viene en el enlace que he usado de Cassandra que es nuevo y que no tiene que ver con la Cassandra instalada de forma local, esto lo he hecho para que se vea como montar Cassandra tanto de forma local como con docker-compose.

¿QUÉ ES AxonOPS Y PARA QUÉ SIRVE?

AxonOPS es una plataforma de monitoreo y gestión para bases de datos distribuidas como Apache Cassandra en mi caso, diseñada para facilitar la administración de clusters. Ofrece una interfaz intuitiva que permite a los administradores supervisar el **estado del cluster en tiempo real, ver métricas de rendimiento, gestionar alertas y realizar tareas de mantenimiento como la creación de backups y restauraciones**. Además, proporciona **acceso a logs detallados**, facilita la visualización del comportamiento del sistema y permite realizar configuraciones personalizadas para optimizar el rendimiento y la estabilidad de los nodos. **Con AxonOPS, la administración de Cassandra se simplifica, mejorando la visibilidad y el control de los entornos productivos.**

Licencia y Open Source:

AxonOps **no es completamente open-source**.

Axon Server Community Edition, que es una parte clave del ecosistema AxonOps, es **open-source y gratuita para proyectos pequeños y pruebas**.

La versión **Enterprise Edition** de Axon Server, que incluye características adicionales como **la escalabilidad avanzada y clustering multi-nodo**, **no es open-source** y requiere una licencia comercial.

AxonOps, en su versión open-source, proporciona un conjunto de herramientas para el monitoreo y gestión de bases de datos distribuidas como Apache Cassandra. **Permite supervisar el estado de los nodos y clusters en tiempo real, acceder a métricas clave de rendimiento, configurar alertas, visualizar logs y eventos, y gestionar configuraciones del sistema a través de una interfaz gráfica**. También soporta la creación de backups y restauraciones básicas, y facilita la escalabilidad horizontal para entornos grandes. Aunque

la versión open-source ofrece funcionalidades completas de monitoreo y gestión, **algunas características avanzadas como optimización de rendimiento y soporte para clústeres más complejos requieren una versión comercial.**

EXPLICACIÓN DE ARQUITECTURA MONTADA

Para montar tras investigar bastante he usado el docker-compose.yml del siguiente enlace:

<https://raw.githubusercontent.com/axonops/axonops-cassandra-dev-cluster/main/docker-compose.yml>

En este entorno, se ha configurado un stack de servicios utilizando Docker Compose con varios contenedores interconectados. El servicio **Elasticsearch** se ejecuta en un contenedor único configurado como un nodo de un clúster, con una configuración optimizada para la memoria (256MB de heap). Se utiliza para almacenar y buscar datos relacionados con los logs y métricas de los servicios que forman el stack.

El **Axon Server** es un contenedor que depende de Elasticsearch y actúa como el servidor central de **AxonOPS**, que es el motor de monitoreo y gestión de Cassandra. Este servicio se conecta a Elasticsearch para almacenar y acceder a los datos de monitoreo.

El servicio **Axon Dash** proporciona una interfaz gráfica en el puerto 3000, donde se pueden visualizar las métricas y el estado del clúster Cassandra, y acceder a la información centralizada de AxonOPS.

El stack también incluye tres nodos de **Cassandra** (cassandra-0, cassandra-1 y cassandra-2), cada uno configurado con sus respectivos puertos de transporte nativos (9042, 9043 y 9044). Estos nodos están conectados entre sí, formando un clúster de Cassandra. Además, cada nodo tiene una configuración optimizada de memoria y está monitoreado por **AxonOPS** para asegurar la disponibilidad y rendimiento. Cada nodo utiliza un volumen persistente para almacenar sus datos y logs, y los contenedores realizan verificaciones periódicas de salud para garantizar su funcionamiento adecuado.

Este entorno proporciona una solución robusta para gestionar y monitorear un clúster Cassandra, con la integración de Elasticsearch y AxonOPS para el monitoreo y análisis centralizado.

1º. Primero he obtenido el docker-compose mediante un curl

```
root@moria:~# curl -O https://raw.githubusercontent.com/axonops/axonops-cassandra-dev-cluster/main/docker-compose.yml
% Total    % Received % Xferd  Average Speed   Time     Time      Current
                                         Dload  Upload   Total   Spent    Left  Speed
100  4252  100  4252    0     0  13736      0  --:--:-- --:--:-- 13760
root@moria:~#
```

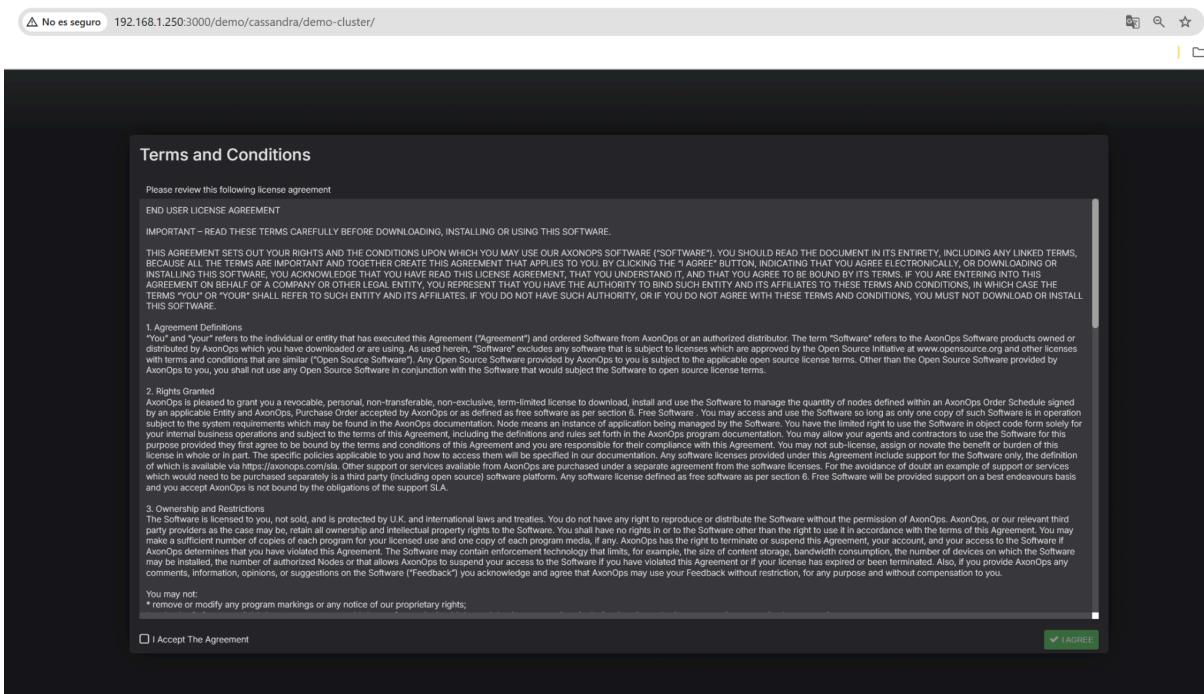
2º. Creamos un stack para montar el compose



3º. Montamos el stack y comprobamos que se ha montado de forma correcta.

```
root@moria:~/stack/cassandra# docker compose up -d
[+] Running 11/11
  ✓ Network cassandra_default Created
  ✓ Container cassandra-elasticsearch-1 Healthy
  ✓ Container cassandra-cassandra-0-1 Healthy
  ✓ Container cassandra-cassandra-0-2 Healthy
  ✓ Container cassandra-cassandra-1-1 Healthy
  ✓ Container cassandra-cassandra-2-1 Started
  ✓ Container cassandra-axon-dash-1 Started
Footprint(moria):~/stack/cassandra# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
83ca1e6f1b59        registry.axonops.com/axonops-public:axon-dash:latest   "/bin/sh -c 'sed -i .."   3 minutes ago       Up 2 minutes (healthy)   0.0.0.0:3000->3000/tcp
aeed071a0709        registry.axonops.com/axonops-public:docker/cassandra:5.0   "/axonops-entrypoint.."  3 minutes ago       Up About a minute (health: starting)  7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp, 0.0.0.0
dc84a40f0f46        registry.axonops.com/axonops-public:axon-server:latest   "/axonops-entrypoint.."  3 minutes ago       Up 2 minutes (healthy)   7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp, 0.0.0.0
t0043342949         /tcp        0.0.0.0:9043->cassandra-cassandra-1-1           "/axonops-entrypoint.."  3 minutes ago       Up 2 minutes (healthy)   7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp, 0.0.0.0
8054342d377b        registry.axonops.com/axonops-public:axonops-docker/axon-server:latest
                      cassandra-axon-server-1
aee4ced0217a        docker.elastic.co/elasticsearch/elasticsearch:7.17.24
                      cassandra-elasticsearch-1
f10574d039da        registry.axonops.com/axonops-public:axonops-docker/cassandra:5.0
                      cassandra-cassandra-0-1
2/tcp, :::9042->9042/tcp
root@moria:~/stack/cassandra#
```

4º. Accedemos mediante el navegador puerto 3000



Una vez accedo como se puede comprobar te enseña la arquitectura/ clúster de Cassandra; además se puede apreciar en el lateral que te permite muchas opciones que profundizaré un poco más adelante.

The screenshot shows the AxonOps interface for a Cassandra cluster named 'demo-cluster'. On the left, there's a navigation sidebar with various monitoring and configuration options. The main area is titled 'Cluster demo-cluster Overview' and features two views: 'GRAPH VIEW' and 'LIST VIEW'. In the GRAPH VIEW, three nodes are shown as green circles with lines connecting them. Each node has a callout box with its details: node 172.18.0.5 (Cassandra version: 5.0.2, Data center: dc1, Rack: rack1, Mode: NORMAL), node 172.18.0.3 (Cassandra version: 5.0.2, Data center: dc1, Rack: rack0, Mode: NORMAL), and node 172.18.0.7 (Cassandra version: 5.0.2, Data center: dc1, Rack: rack2, Mode: NORMAL). The LIST VIEW below shows a table with columns: DC, Filter..., Rack, Filter..., Node, Service Checks, Alerts, Endpoint, Filter..., Mode, Tasks, Agent, Agent Version, Cassandra Version. The data from the table matches the nodes shown in the graph.

DC	Filter...	Rack	Filter...	Node	Service Checks	Alerts	Endpoint	Filter...	Mode	Tasks	Agent	Agent Version	Cassandra Version
dc1		rack0		172.18.0.3	●	●	172.18.0.3		NORMAL	-	●	1.0.121	5.0.2
dc1		rack1		172.18.0.5	●	●	172.18.0.5		NORMAL	-	●	1.0.121	5.0.2
dc1		rack2		172.18.0.7	●	●	172.18.0.7		NORMAL	-	●	1.0.121	5.0.2

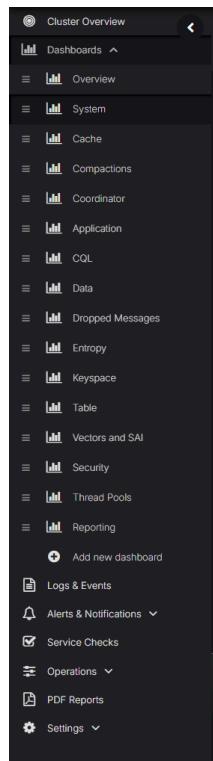
This screenshot shows the same AxonOps interface as the previous one, but in the 'LIST VIEW' mode. It displays a table with the following data:

DC	Filter...	Rack	Filter...	Node	Service Checks	Alerts	Endpoint	Filter...	Mode	Tasks	Agent	Agent Version	Cassandra Version
dc1		rack0		172.18.0.3	●	●	172.18.0.3		NORMAL	-	●	1.0.121	5.0.2
dc1		rack1		172.18.0.5	●	●	172.18.0.5		NORMAL	-	●	1.0.121	5.0.2
dc1		rack2		172.18.0.7	●	●	172.18.0.7		NORMAL	-	●	1.0.121	5.0.2

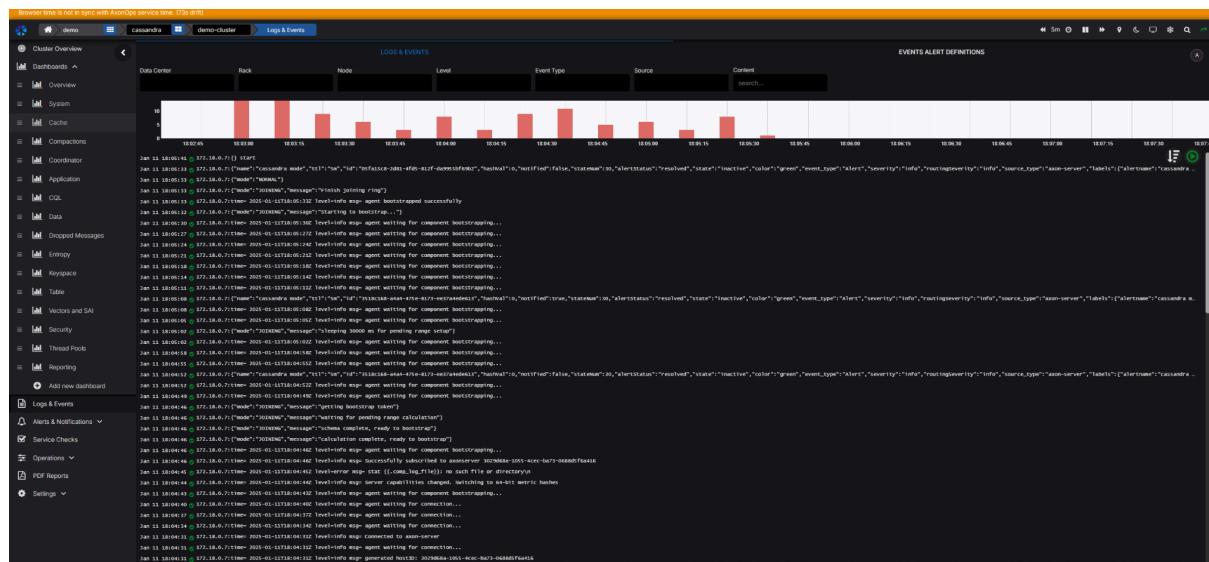
5º. Una vez estoy dentro en el apartado **Dashboards** te permite crear gráficos y verlos, en este caso estoy viendo un dashboard por defecto del sistema.

This screenshot shows the AxonOps interface in the 'System' dashboard. The left sidebar has a 'Dashboards' section with a 'System' option selected. The main area contains four graphs: 'CPU And Load' (showing CPU usage per host and load average over 15m), 'Disk Usage Detail' (with tabs for Data Center, Rack, Node, Mount Point, Partition, and Interface), 'Memory Statistics' (with tabs for Data Center, Rack, Node, and Interface), and 'Network Statistics' (with tabs for Data Center, Rack, Node, and Interface). There is also a 'CPU Usage Detail' section with a 'CPU usage per host' chart. At the bottom, there's a search bar and a table for event logs.

6º. En el lateral izquierdo se pueden ver todas estas opciones (que iré viendo una a una)



7º. Una de esas opciones es la visualización de logs que se pueden ver tanto escritos como de forma gráfica.



8º. También te permite crear alertas, estas se pueden crear sobre eventos que ocurren en el sistema de cada nodo o de aplicaciones que tengas en estos.

Status	DC	Rack	Node	Filter...	Message
green	dc1	rack2	17.18.0.7		-
green	dc1	rack1	17.18.0.5		-
green	dc1	rack0	17.18.0.3		-

9º. Además cabe destacar que te permite crear backups y restaurarlos como se puede ver en las imágenes.

Operations

- Repairs
- Rolling Restart
- Backups
- Restore

Cassandra Backups

Schedules

Tag: Filtar... Data Centers: Keypaces: Tables: Last Run: Next Run:

Backup type: Immediate Simple Schedule Cron Schedule

Data Center (mandatory): dc1

Keyspace (optional): Select a keyspace

Tables (optional): Add tables

Local Snapshot Retention: 10d

Backup timeout (backup job will be aborted if it runs for longer than this time): 10h

5m, 2h, 1d ...

Remote Backups: Remote Backups

CREATE BACKUP **TEST**

PAUSE BACKUP SCHEDULER

Cassandra Restore

SNAPSHOT BACKUP RESTORE POINT-IN-TIME RESTORE

Available Backups for Restore

Actions	Creation Time	Time End	Tag	Filter...	Local Status	Remote Status	Data Centers	Nodes	Tables	Status
RESTORE	2025/01/11 18:14:36	2025/01/11 18:14:37	-		ok	no remote	dc1	All	All	<input checked="" type="checkbox"/>

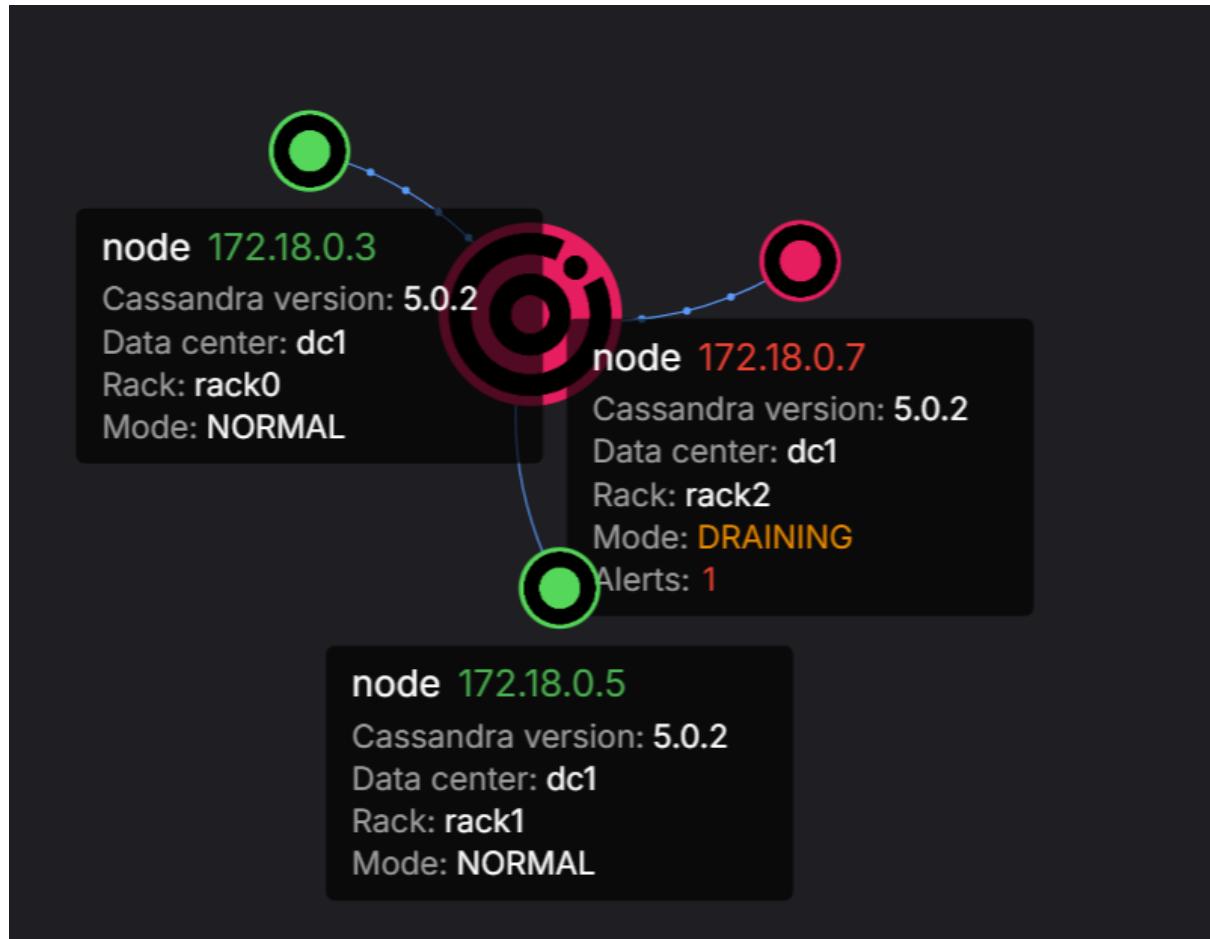
Backup Restoration History

Time	Status	Duration	User	DCs	Racks	Nodes
------	--------	----------	------	-----	-------	-------

10º. También por concluir este apartado ya que esta herramienta al ser una prueba tampoco puedo llegar a enseñar todas las opciones que tiene su función de pago, enseñaré una prueba sobre la caída de un nodo que como se apreciará en la imagen cambiará de color y te alertará

sobre esta caída.

```
root@moria:~/stack/cassandra# docker ps
CONTAINER ID IMAGE NAMES COMMAND CREATED STATUS PORTS
83ca186f859 registry.axonops-public/axonops-docker/axon-dash:latest " /bin/sh -c 'sed -i _" 35 minutes ago Up 34 minutes (healthy) 0.0.0.0:3000->3000/tcp, :::3
000->3000/tcp
aecd0711a7d9 registry.axonops.com/axonops-public/axonops-docker/cassandra:5.0 cassandra-axon-dash-1 "/bin/sh -c 'sed -i _" 35 minutes ago Up 34 minutes (healthy) 7000-7001/tcp, 7199/tcp, 904
2/tcp, 9160/tcp, 0.0.0.0:9044->9044/tcp, :::9044->9044/tcp
cassandra-cassandra-2-1 "/axonops-entrypoint_"
dc6eaef0fe8e registry.axonops.com/axonops-public/axonops-docker/cassandra:5.0 cassandra-axon-dash-1-1 "/axonops-entrypoint_"
2/tcp, 9160/tcp, 0.0.0.0:9043->9043/tcp, :::9043->9043/tcp
cassandra-cassandra-1-1 "/usr/share/axonops_"
8054426377b registry.axonops.com/axonops-public/axonops-docker/cassandra-server:latest cassandra-axon-server-1 "/bin/tini -- /usr/l_"
aeec4eda217a docker.elastic.co/elasticsearch/elasticsearch:7.17.24 cassandra-elasticsearch-search-1 "/bin/tini -- /usr/l_"
f10574d039da registry.axonops.com/axonops-public/axonops-docker/cassandra:5.0 cassandra-cassandra-0-1 "/axonops-entrypoint_"
0/tcp, 0.0.0.0:9042->9042/tcp, :::9042->9042/tcp
root@moria:~/stack/cassandra# docker stop cassandra-cassandra-2-1
cassandra-cassandra-2-1
root@moria:~/stack/cassandra#
```



También crea una alerta al apagar uno de los nodos

Time	Host Id	Message	Actions
2021-11-18T13:33	172.18.0.7	Agent connectivity unstable	Acknowledge

Active alerts count: 1
Time since last alert: 1 minute

CREACIÓN DE USUARIOS Y PERMISOS

La gestión de usuarios y permisos se centra en garantizar la seguridad de los datos mediante autenticación y autorización.

AUTENTICACIÓN Y AUTORIZACIÓN

Dentro del fichero `/etc/cassandra/cassandra.yaml` se pueden hallar los siguientes parámetros que sirven para restringir o proporcionar permisos de ejecución o realización de operaciones además de permitir la conexión a Cassandra.

- Para autenticar se puede usar el parámetro **PasswordAuthenticator** que sirve para gestionar usuarios y contraseñas, por defecto viene el parámetro **AllowAllAuthenticator** que permite acceder a todo el mundo.
- Para autorizar se puede usar el parámetro **CassandraAuthorizer** que controla qué operaciones puede realizar un usuario, por defecto viene el parámetro **AllowAllAuthorizer** que permite realizar todo tipo de ejecuciones.

```
# Using PasswordAuthenticator, C
authenticator: AllowAllAuthenticator

# Authorization backend, implementing
# Out of the box, Cassandra provides
# CassandraAuthorizer}.
#
# - AllowAllAuthorizer allows any act
# - CassandraAuthorizer stores permis
#   increase system_auth keyspace rep
authorizer: AllowAllAuthorizer
```

CREACIÓN DE USUARIOS

Para crear usuarios, necesitas ser un **superusuario** (como el **usuario cassandra** por defecto).

Crear un usuario

1º. Tendremos que cambiar el modo de **autorización y autenticación**

```
# If using PasswordAuthenticator, CassandraRo
authenticator: PasswordAuthenticator

# Authorization backend, implementing IAuthoriz
# Out of the box, Cassandra provides org.apache
# CassandraAuthorizer}.

#
# - AllowAllAuthorizer allows any action to any
# - CassandraAuthorizer stores permissions in s
#   increase system_auth keyspace replication f
authorizer: CassandraAuthorizer
```

```
root@Tifa:~# sudo systemctl restart cassandra
```

2º. Entraremos con el usuario **cassandra** y crearemos un usuario **superusuario** y otro **no superusuario**

```
root@Tifa:~# cqlsh 138.199.149.188 -u cassandra
Password:
Connected to Test Cluster at 138.199.149.188:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
```

3º. Crearemos los usuarios

```
cassandra@cqlsh> CREATE ROLE admin_user WITH PASSWORD = 'securePass123' AND SUP
ERUSER = true AND LOGIN = true;
cassandra@cqlsh> CREATE ROLE regular_user WITH PASSWORD = 'userPass123' AND SUP
ERUSER = false AND LOGIN = true;
cassandra@cqlsh>
```

Ver usuarios creados

Para ello tendremos que listar los roles que son realmente los usuarios que hay en la BD.

```
use HELP for help.
cassandra@cqlsh> LIST ROLES;

  role      | super | login | options | datacenters
-----+-----+-----+-----+-----+
admin_user | True | True | {} | ALL
cassandra  | True | True | {} | ALL
regular_user | False | True | {} | ALL

(3 rows)
cassandra@cqlsh> █
```

Modificar un usuario

1º. Para que se vea que solo puedo modificar un usuario con un usuario superusuario entraré con el usuario regular y probaré a cambiar tanto la contraseña del usuario regular como el del administrador

```
regular_user@cqlsh> ALTER USER regular_user WITH PASSWORD 'new_password';
regular_user@cqlsh> ALTER USER admin_user WITH PASSWORD 'new_password';
Unauthorized: Error from server: code=2100 [Unauthorized] message="User regular
_user does not have sufficient privileges to perform the requested operation"
regular_user@cqlsh> █
```

2º. Probaré a cambiarla con el usuario admin que tiene el rol superusuario

```
root@Tifa:~# cqlsh 138.199.149.188 -u admin_user
Password:
Connected to Test Cluster at 138.199.149.188:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
admin_user@cqlsh> ALTER USER regular_user WITH PASSWORD 'new_password';
admin_user@cqlsh> ALTER USER admin_user WITH PASSWORD 'new_password';
admin_user@cqlsh> █
```

Eliminar un usuario

Intentaré eliminar con el usuario regular, pero solo me dejará con un usuario **superusuario**, para ello usaré el comando **drop user**.

```
regular_user@cqlsh> DROP USER regular_user;
Unauthorized: Error from server: code=2100 [Unauthorized] message="User regular
_user does not have sufficient privileges to perform the requested operation"
regular_user@cqlsh> █
```

```
admin_user@cqlsh> DROP USER regular_user;
admin_user@cqlsh> list roles;

role      | super | login | options | datacenters
-----+-----+-----+-----+
admin_user | True  | True  | {}     | ALL
cassandra  | True  | True  | {}     | ALL

(2 rows)
admin_user@cqlsh> █
```

ASIGNACIÓN DE PERMISOS

Los permisos en Cassandra se gestionan a nivel de recursos: **keyspaces**, **tablas**, y **funciones**.

Permisos disponibles:

- **ALL PERMISSIONS**: Otorga todos los permisos.
- **ALTER**: Permite modificar esquemas.
- **AUTHORIZE**: Permite conceder permisos a otros usuarios.
- **DROP**: Permite eliminar recursos.
- **MODIFY**: Permite realizar operaciones de escritura (INSERT, UPDATE, DELETE).
- **SELECT**: Permite realizar consultas de lectura.

Otorgar permisos

Primero le daré permisos de select en el **keyspace my_keyspace** al usuario **regular**, para ello usaré el comando **grant**

```
admin_user@cqlsh> GRANT SELECT ON KEYSPACE my_keyspace TO regular_user;
```

Luego de modificación

```
admin_user@cqlsh> GRANT MODIFY ON KEYSPACE my_keyspace TO regular_user;
admin_user@cqlsh> █
```

Revocar permisos

Para quitar los permisos usaremos el comando **revoke**, en este caso los quitaré de modificar.

```
admin_user@cqlsh> REVOKE MODIFY ON KEYSPACE my_keyspace FROM regular_user;
```

Ver permisos

Comprobaremos los permisos tanto del usuario **regular** que solo debería de tener permiso **select** del keyspace **my_keyspace** y también veremos al usuario **cassandra** que tendrá **all permission de todo** (es decir es super usuario y tiene permisos para realizar todo tipo de tareas).

```
admin_user@cqlsh> LIST ALL PERMISSIONS OF regular_user;

role      | username      | resource                  | permission
-----+-----+-----+-----+
regular_user | regular_user | <keyspace my_keyspace> |      SELECT
(1 rows)
admin_user@cqlsh> LIST ALL PERMISSIONS OF cassandra;

role      | username      | resource                  | permission
-----+-----+-----+-----+
cassandra | cassandra    | <role admin_user> |      ALTER
cassandra | cassandra    | <role admin_user> |      DROP
cassandra | cassandra    | <role admin_user> |      AUTHORIZE
cassandra | cassandra    | <role admin_user> |      DESCRIBE
(4 rows)
admin_user@cqlsh> █
```

SEGURIDAD. DIRECTIVAS RELEVANTES DEL SGBD. HERRAMIENTAS DE BACKUPS.

En cuanto a la seguridad podemos mencionar:

AUTENTICACIÓN Y AUTORIZACIÓN

En cuanto a este apartado como mencioné anteriormente el apartado de creación de usuarios hay dos parámetros que se pueden usar en el fichero **/etc/cassandra/cassandra.yaml** que son **PasswordAuthenticator** y **Cassandra Authorizer** que se puede

- Para autenticar se puede usar el parámetro **PasswordAuthenticator** que sirve para gestionar usuarios y contraseñas, por defecto viene el parámetro **AllowAllAuthenticator** que permite acceder a todo el mundo.
- Para autorizar se puede usar el parámetro **CassandraAuthorizer** que controla qué operaciones puede realizar un usuario, por defecto viene el parámetro **AllowAllAuthorizer** que permite realizar todo tipo de ejecuciones.

Para verificar que este funciona lo comprobaremos accediendo por ejemplo con el usuario **Cassandra** que viene por defecto, sin esos parámetros no podríamos crear un usuario pero al tenerlos activos si podremos.

```
root@EzioCloud:~# cqlsh -u cassandra
Password:
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> CREATE USER nombre_usuario WITH PASSWORD 'contraseña' SUPERUSER;
Unauthorized: Error from server: code=2100 [Unauthorized] message="Only superusers can create a role with superuser status"
cassandra@cqlsh>
```

Por lo que deberemos de cambiar los siguientes parámetros

```
# If using PasswordAuthenticator, CassandraRole
authenticator: PasswordAuthenticator

# Authorization backend, implementing IAuthoriz
# Out of the box, Cassandra provides org.apache
# CassandraAuthorizer}.
#
# - AllowAllAuthorizer allows any action to any
# - CassandraAuthorizer stores permissions in s
#   increase system_auth keyspace replication f
authorizer: CassandraAuthorizer
```

```
root@EzioCloud:~# sudo nano /etc/cassandra/cassandra.yaml
root@EzioCloud:~# sudo systemctl restart cassandra
```

Nos meteremos en cassandra y comprobaremos que ya si podemos

```
root@EzioCloud:~# cqlsh -u cassandra
Password:
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> CREATE USER Nico WITH PASSWORD 'bolson' SUPERUSER;
cassandra@cqlsh> list roles;

role      | super | login | options | datacenters
-----+-----+-----+-----+-----+
  Nico   |  True |  True |    {}   |      ALL
cassandra |  True |  True |    {}   |      ALL

(2 rows)
cassandra@cqlsh>
```

PERMISOS

Es algo que ya mencioné anteriormente, es algo bastante necesario y muy importante ya que según las necesidades de cada usuario puede usarse distintos permisos.

Los permisos en Cassandra se gestionan a nivel de recursos: **keyspaces**, **tablas**, y **funciones**.

Permisos disponibles:

- **ALL PERMISSIONS:** Otorga todos los permisos.
- **ALTER:** Permite modificar esquemas.
- **AUTHORIZE:** Permite conceder permisos a otros usuarios.
- **DROP:** Permite eliminar recursos.
- **MODIFY:** Permite realizar operaciones de escritura (INSERT, UPDATE, DELETE).
- **SELECT:** Permite realizar consultas de lectura.

Además cabe mencionar que hay otra opción de permisos que es ser **Superuser o no**

Diferencias clave

Característica	Superusuari	No superusuari
----------------	-------------	----------------

Alcance de permisos	Global (todos los keyspaces y tablas)	Limitado a permisos asignados
Gestión de usuarios	Puede crear y administrar usuarios	No puede administrar usuarios
Gestión de permisos	Puede conceder/revocar permisos	No puede otorgar permisos
Acceso predeterminado	Total, sin restricciones	Restringido según configuración
Propósito principal	Administración del sistema	Uso operativo o de aplicación

Para comprobar que un superusuario puede crear otro usuario o puede eliminar un usuario cualquiera además de eliminar una tabla o dar permisos a diferencia de un no superusuario que solo tiene los permisos que tú le asignes haré una demostración:

```
root@EzioCloud:~# cqlsh -u Nico
Password:
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
Nico@cqlsh> CREATE USER nico2 WITH PASSWORD 'bolson';
Nico@cqlsh> list roles;

role      | super | login | options | datacenters
-----+-----+-----+-----+-----+
  Nico  |  True |  True |    {}    |      ALL
cassandra |  True |  True |    {}    |      ALL
  nico2 | False |  True |    {}    |      ALL

(3 rows)
Nico@cqlsh>
```

Una vez creado el usuario se verá que no tiene permisos para crear una base de datos, por lo que tendrá que ser creada por el **superusuario**

```
root@EzioCloud:~# cqlsh -u nico2
Password:
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
nico2@cqlsh> CREATE KEYSPACE mi_keyspace WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};
Unauthorized: Error from server: code=2100 [Unauthorized] message="User nico2 has no CREATE permission on <all keyspaces> or any of its parents"
nico2@cqlsh>

Nico@cqlsh> CREATE KEYSPACE mi_keyspace WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};

Nico@cqlsh> describe keyspaces;
          name           strategy   replicas
mi_keyspace      SimpleStrategy      3
system           SimpleStrategy      1
system_auth       SimpleStrategy      1
system_distributed SimpleStrategy      1
system_schema     SimpleStrategy      1
system_traces     SimpleStrategy      1
system_views      SimpleStrategy      1
system_virtual_schema SimpleStrategy      1
Nico@cqlsh>
```

CIFRADO DE DATOS

En Cassandra, el **cifrado de datos** se configura para proteger tanto los datos en reposo (almacenados en disco) como los datos en tránsito (que circulan entre nodos y clientes). Esto se controla principalmente mediante las siguientes directivas en el archivo de configuración **cassandra.yaml**

Cifrado de datos en tránsito

Se utiliza para asegurar las conexiones entre clientes y nodos, así como entre los nodos del clúster. Las directivas importantes son:

- **server_encryption_options**: Configura el cifrado entre nodos del clúster.
- **client_encryption_options**: Configura el cifrado entre clientes y nodos.
- **enabled**: Activa o desactiva el cifrado (true o false).
- **optional**: Permite conexiones sin cifrar si no todas las conexiones admiten TLS.
- **keystore y truststore**: Rutas a los almacenes de claves y certificados (JKS) para configurar TLS.

```
GNU nano 8.3           /etc/cassandra/cassandra.yaml *

client_encryption_options:
  enabled: true
  optional: false
  keystore: conf/.keystore
  keystore_password: cassandra
  truststore: conf/.truststore
  truststore_password: cassandra
```

Cifrado de datos en reposo

Se asegura de que los datos almacenados en el disco (SSTables) estén cifrados. Esto se habilita mediante **cifrado en los almacenes de claves (Keyspace encryption)**, utilizando un **Key Management System (KMS)** integrado.

- **encryption_options** (para keyspaces): Configura el algoritmo de cifrado, clave y proveedor.
- **transparent_data_encryption**: Activa el cifrado automático en los SSTables.

```
GNU nano 8.3           /etc/cassandra/cassandra.yaml *

transparent_data_encryption:
  enabled: true
  key_provider:
    - class_name: org.apache.cassandra.security.JKSKeyProvider
      parameters:
        keystore: conf/.keystore
        keystore_password: cassandra
        store_type: JKS
```

¿Para qué sirve?

El cifrado en Cassandra protege datos sensibles contra accesos no autorizados, tanto en movimiento como en almacenamiento. El **cifrado en tránsito** evita que los datos sean interceptados entre nodos o clientes, mientras que el **cifrado en reposo** asegura que los datos no puedan leerse si el almacenamiento físico es comprometido.

Pondré un ejemplo de cifrado en tránsito y verificaré que se están cifrando las conexiones entre los nodos (en mi caso solo tengo 1 nodo por lo que se verá de forma más óptima cuando implemente un clúster)

1º. Generaré un cifrado, para ello crearé una clave privada y pública, además de un certificado para cada nodo

2º. Crearé un **keystore**, para ello primero crearé un archivo .p12 y luego ejecutaré un **keytool**.

```
root@EzioCloud:~# openssl pkcs12 -export -in cassandra.crt -inkey cassandra.key  
-out cassandra.p12
```

Enter pass phrase for cassandra key:

Enter pass phrase for
Enter Export Password:

Verifying - Enter Export Password:

```
[root@FzioCloud: ~]# ls
```

```
cassandra_crt cassandra_key cassandra_p12 scriptfirewall.sh
```

cassandra:~ cass
root@EzioCloud:~#

```
root@EzioCloud:~# keytool -importkeystore -srckeystore cassandra.p12 -srcstoretype PKCS12 -destkeystore cassandra.keystore -deststoretype JKS
Importing keystore cassandra.p12 to cassandra.keystore...
Enter destination keystore password:
Re-enter new password:
Enter source keystore password:
Entry for alias 1 successfully imported.
Import command completed: 1 entries successfully imported, 0 entries failed or cancelled

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -importkeystore -srckeystore cassandra.keystore -destkeystore cassandra.keystore -deststoretype pkcs12".
root@EzioCloud:~#
```

3º. Crearé un **trustore** (almacén de confianza)

El truststore es donde Cassandra almacena los certificados de las entidades de confianza (como las CA que han firmado los certificados).

```
root@EzioCloud:~# keytool -import -alias cassandra-ca -file cassandra.crt -key
tore cassandra.truststore
Enter keystore password:
Re-enter new password:
Owner: O=Internet Widgit Pty Ltd, ST=Some-State, C=AU
Issuer: O=Internet Widgit Pty Ltd, ST=Some-State, C=AU
Serial number: 6f76dfcd9b1ca2d2f42709c5d1c06400326302f9
Valid from: Mon Jan 13 21:18:05 UTC 2025 until: Tue Jan 13 21:18:05 UTC 2026
Certificate fingerprints:
    SHA1: E5:32:84:DD:87:D5:D8:B7:9E:06:E8:2D:F7:9D:9F:EB:5A:AD:9B:D5
    SHA256: 1A:79:52:30:3B:0C:C1:3A:0E:A1:69:88:C0:8B:27:69:D9:83:F9:BA:0
:83:B9:F9:65:5A:0F:91:A9:2E:51:9F
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: FF 35 64 2B B9 08 F3 1A 8E 6E FA E8 3B 91 DD 22 .5d+....n..;..
0010: AA 52 A9 52 .R.R
]
]

#2: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: FF 35 64 2B B9 08 F3 1A 8E 6E FA E8 3B 91 DD 22 .5d+....n..;..
0010: AA 52 A9 52 .R.R
]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
root@EzioCloud:~#
```

4º. Configuraré **cassandra.yaml** (necesario hacer en cada nodo del clúster aunque en mi caso solo lo haré en el que tengo)

```
GNU nano 7.2          /etc/cassandra/cassandra.yaml *
#   excluded_keyspaces: # comma separated list of keyspaces to exclude from t#
#   excluded_tables: # comma separated list of keyspace.table pairs to exclude#
server_encryption_options:
  # Activar cifrado entre nodos
  internode_encryption: all # Puede ser 'none', 'all', 'dc' o 'rack'

  # Ruta al keystore (con las claves del nodo)
  keystore: /root/cassandra.keystore

  # Contraseña del keystore
  keystore_password: "bolson"

  # Ruta al truststore (con las CA de confianza)
  truststore: /root/cassandra.truststore

  # Contraseña del truststore
  truststore_password: "bolson"

  # Algoritmo de cifrado
  cipher_suites: TLS_RSA_WITH_AES_128_CBC_SHA256
```

5º. Reiniciamos Cassandra

```
root@EzioCloud:~# sudo systemctl restart cassandra
root@EzioCloud:~# sudo systemctl status cassandra
● cassandra.service - LSB: distributed storage system for structured data
  Loaded: loaded (/etc/init.d/cassandra; generated)
  Active: active (exited) since Mon 2025-01-13 21:29:40 UTC; 9s ago
  Invocation: bbfcd33d3db642dabd87a62a8e49fd3
    Docs: man:systemd-sysv-generator(8)
  Process: 28030 ExecStart=/etc/init.d/cassandra start (code=exited, status=>
  Mem peak: 1.1G
  CPU: 4.162s

Jan 13 21:29:40 EzioCloud systemd[1]: Starting cassandra.service - LSB: distribu
Jan 13 21:29:40 EzioCloud systemd[1]: Started cassandra.service - LSB: distribu
[lines 1-11/11 (END)]
```

6º. Añadimos lo siguiente en /root/.cassandra/cqlshrc

```
GNU nano 7.2                               /root/.cassandra/cqlsh *
[ssl]
certfile = /root/cassandra.crt
keyfile = /root/cassandra.key
ca_certs = /root/cassandra.crt
```

Y al reiniciar nos debería de dejar ejecutar el comando nodetool status (en mi caso no me ha dejado he tenido errores para poder llevarlo a cabo pero he querido dejarlo porque es algo que aporta bastante seguridad y es muy recomendable)

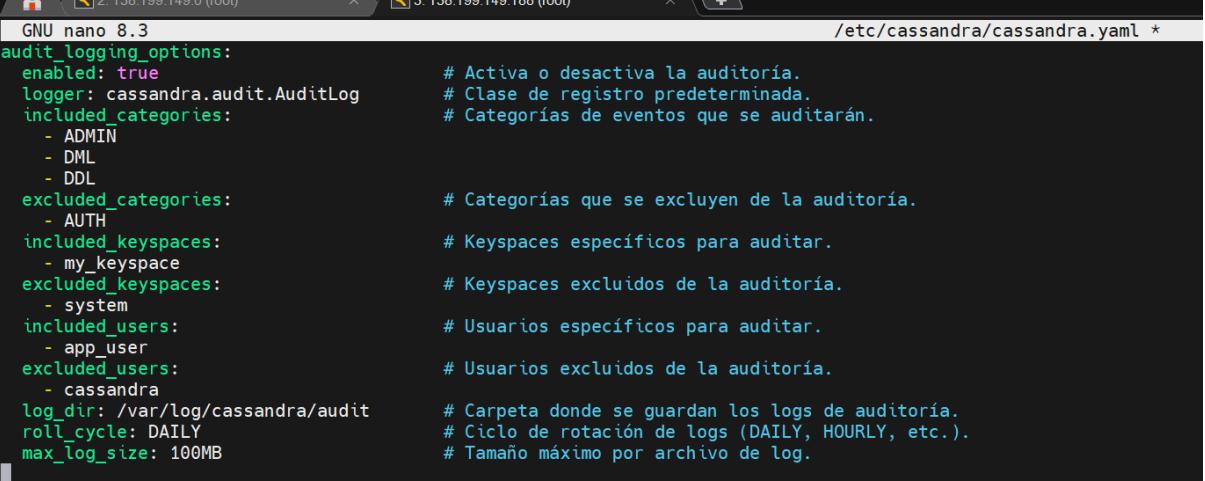
```
root@EzioCloud:~# nodetool status
nodetool: Failed to connect to '127.0.0.1:7199' - ConnectException: 'Connection refused (Connection refused)'.
```

AUDITORÍAS

Las auditorías en Cassandra son esenciales para rastrear y registrar las actividades realizadas en el sistema, como operaciones de lectura, escritura, cambios de configuración, y accesos de usuarios. Te ayuda con accesos no autorizados o actividades sospechosas.

- **enabled:** Activa/desactiva la funcionalidad de auditoría.
- **included_categories y excluded_categories:**
 - **ADMIN:** Acciones administrativas, como gestión de roles o nodos.
 - **DML:** Operaciones de datos (SELECT, INSERT, UPDATE, DELETE).
 - **DDL:** Cambios en la definición del esquema (crear tablas, keyspaces, índices).
 - **AUTH:** Eventos relacionados con autenticación (por defecto excluidos).
- **log_dir:** Especifica dónde se almacenan los registros de auditoría.
- **roll_cycle:** Define con qué frecuencia se rota el archivo de log.
- **included_keyspaces y excluded_keyspaces:** Permite restringir la auditoría a keyspaces específicos.
- **included_users y excluded_users:** Define qué usuarios están sujetos o exentos de auditoría.

Y por ejemplo un uso de estos parámetros dentro del fichero de configuración:



```

GNU nano 8.3
/etc/cassandra/cassandra.yaml *

audit_logging_options:
  enabled: true          # Activa o desactiva la auditoría.
  logger: cassandra.audit.AuditLog   # Clase de registro predeterminada.
  included_categories:
    - ADMIN
    - DML
    - DDL
  excluded_categories:
    - AUTH
  included_keyspaces:
    - my_keyspace
  excluded_keyspaces:
    - system
  included_users:
    - app_user
  excluded_users:
    - cassandra
  log_dir: /var/log/cassandra/audit
  roll_cycle: DAILY        # Ciclo de rotación de logs (DAILY, HOURLY, etc.).
  max_log_size: 100MB      # Tamaño máximo por archivo de log.

```

Estas auditorías son usadas por herramientas como **Nodetool**, **Grafana**, **AxonOPS**, **Apache SkyWalking**...

Son usadas para monitorizar el acceso de Usuarios, Cambios de configuración...

Como los logs y las directivas van en menor o mayor medida ligados, los usaré en el apartado de monitorización y logs de cassandra.

DIRECTIVAS RELEVANTES DEL SGBD

En Cassandra, las **directivas relevantes** son configuraciones definidas principalmente en el archivo **cassandra.yaml**, y afectan el comportamiento del sistema de gestión de bases de datos (SGBD). Estas directivas se dividen en varias categorías que incluyen gestión de **nodos**, **clúster**, **almacenamiento**, **seguridad** y **rendimiento**.

Directiva	Descripción	Ejemplo
cluster_name	Nombre del clúster. Debe ser el mismo en todos los nodos del clúster.	cluster_name: 'MiClusterCassandra'
num_tokens	Número de tokens asignados a cada nodo para el balance de datos.	num_tokens: 256

seed_provider	Define los nodos semilla para la formación del clúster.	seed_provider: - class_name: org.apache.cassandra.locator.SimpleSeedProvider parameters: [seeds: "192.168.1.1,192.168.1.2"]
listen_address	Dirección IP en la que el nodo escucha para comunicaciones internas.	listen_address: 192.168.1.10
rpc_address	Dirección IP en la que el nodo escucha para comunicaciones con los clientes.	rpc_address: 0.0.0.0
endpoint_snitch	Determina la topología del clúster (cómo se organiza el rack y el centro de datos).	endpoint_snitch: GossipingPropertyFileSnitch
data_file_directories	Directorios donde se almacenan los datos de Cassandra.	data_file_directories: - /var/lib/cassandra/data
commitlog_directory	Directorio donde se almacenan los logs de commit para asegurar la durabilidad.	commitlog_directory: /var/lib/cassandra/commitlog
saved_caches_directory	Directorio donde se almacenan los archivos de caché de Cassandra.	saved_caches_directory: /var/lib/cassandra/saved_caches
memtable_flush_writers	Número de hilos que se usan para volcar las memtables a disco.	memtable_flush_writers: 2
concurrent_threads	Número de operaciones de lectura concurrentes permitidas.	concurrent_reads: 32
concurrent_writes	Número de operaciones de escritura concurrentes permitidas.	concurrent_writes: 32
authenticator	Configura el mecanismo de	authenticator:

	autenticación.	PasswordAuthenticator
authorizer	Define el mecanismo de autorización de usuarios.	authorizer: CassandraAuthorizer
server_encryption_options	Configura el cifrado entre nodos.	server_encryption_options: internode_encryption: all keystore: conf/.keystore
client_encryption_options	Configura el cifrado entre el cliente y el servidor.	client_encryption_options: enabled: true optional: false
auto_snapshot	Activa la creación automática de snapshots antes de eliminar un keyspace o tabla.	auto snapshot: true

FIREWALL

Es algo muy importante dentro de la protección de nuestra base de datos porque nos permite **asegurar la comunicación dentro del clúster y entre los nodos**. Cassandra tiene ciertos puertos ya mencionados anteriormente que son bastante importantes, para ello usaré una tabla.

Servicio/Protocolo	Puerto	Descripción
Inter-node communication	7000 (TCP)	Puerto utilizado para la comunicación entre nodos dentro del clúster (no cifrado).

Inter-node communication (cifrado)	7001 (TCP)	Puerto utilizado para la comunicación cifrada entre nodos dentro del clúster (cuando internode_encryption está habilitado).
RPC (Cliente a Nodo)	9042 (TCP)	Puerto utilizado para la comunicación entre los clientes y los nodos de Cassandra a través de CQL (Cassandra Query Language).
JMX Management Extensions (Java)	7199 (TCP)	Puerto utilizado para la monitorización mediante JMX, permitiendo que herramientas como JConsole o Prometheus JMX Exporter obtengan métricas del nodo.
Thrift (Obsoleto)	9160 (TCP)	Puerto utilizado por el antiguo protocolo Thrift para la comunicación cliente-servidor. Este puerto está en desuso y se recomienda evitarlo.

Para ello crearé un script para capar los puertos no necesarios y tener más seguridad en mi servidor y también base de datos

```
GNU nano 7.2                                     scriptfirewall.sh *
#!/bin/bash

# Crear la tabla y las cadenas
sudo nft add table inet filter
sudo nft add chain inet filter input { type filter hook input priority 0; policy drop; }
sudo nft add chain inet filter forward { type filter hook forward priority 0; policy drop; }
sudo nft add chain inet filter output { type filter hook output priority 0; policy accept; }

# Permitir tráfico en la interfaz local
sudo nft add rule inet filter input iif lo accept
sudo nft add rule inet filter output oif lo accept

# Permitir tráfico para puertos esenciales (SSH, HTTP, HTTPS, y otros)
sudo nft add rule inet filter input tcp dport 22 accept # SSH
sudo nft add rule inet filter input tcp dport 80 accept # HTTP
sudo nft add rule inet filter input tcp dport 8080 accept # HTTP alternativo
sudo nft add rule inet filter input tcp dport 3000 accept # Usado por aplicaciones
sudo nft add rule inet filter input tcp dport 9000 accept # Usado por herramientas de monitorización
sudo nft add rule inet filter input tcp dport 443 accept # HTTPS
sudo nft add rule inet filter input tcp dport 7000 accept # Cassandra comunicación interna
sudo nft add rule inet filter input tcp dport 7001 accept # Cassandra comunicación cifrada
sudo nft add rule inet filter input tcp dport 9042 accept # CQL (Cassandra)
sudo nft add rule inet filter input tcp dport 7199 accept # JMX (Cassandra)

# Rechazar tráfico en el puerto Thrift (9160), ya que es obsoleto
sudo nft add rule inet filter input tcp dport 9160 reject

# Verificar las reglas
sudo nft list ruleset
```

Lo ejecutaremos y comprobamos si se han aplicado las reglas (al ejecutarse muestra las reglas)

```
root@EzioCloud:~# nano scriptfirewall.sh
root@EzioCloud:~# ./scriptfirewall.sh
table inet filter {
    chain input {
        type filter hook input priority filter; policy drop;
        iif "lo" accept
        tcp dport 22 accept
        tcp dport 80 accept
        tcp dport 8080 accept
        tcp dport 3000 accept
        tcp dport 9000 accept
        tcp dport 443 accept
        tcp dport 7000 accept
        tcp dport 7001 accept
        tcp dport 9042 accept
        tcp dport 7199 accept
        iif "lo" accept
        tcp dport 22 accept
        tcp dport 80 accept
        tcp dport 8080 accept
        tcp dport 3000 accept
        tcp dport 9000 accept
        tcp dport 443 accept
        tcp dport 7000 accept
        tcp dport 7001 accept
        tcp dport 9042 accept
        tcp dport 7199 accept
        tcp dport 9160 reject
    }

    chain forward {
        type filter hook forward priority filter; policy drop;
    }

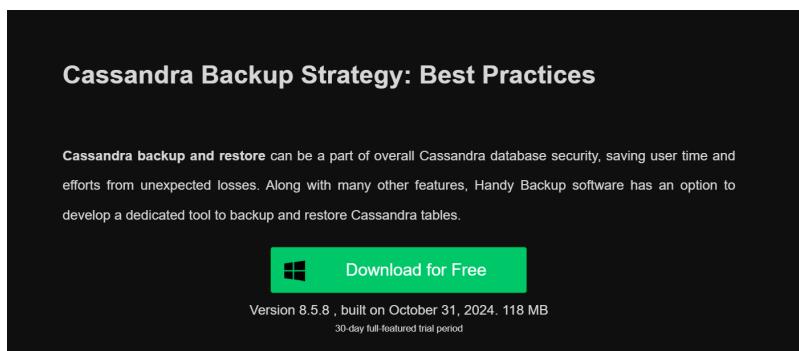
    chain output {
        type filter hook output priority filter; policy accept;
        oif "lo" accept
        oif "lo" accept
    }
}
root@EzioCloud:~#
```

HERRAMIENTAS DE BACKUPS.

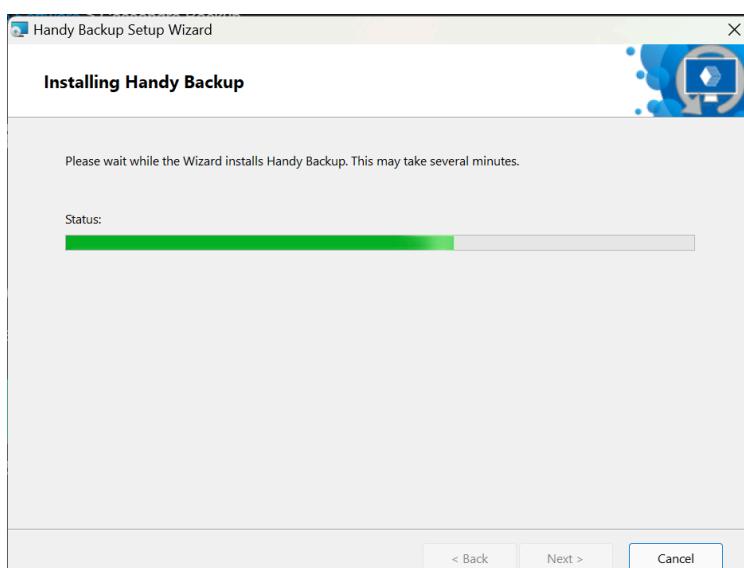
Handy Backup

HandyBackup es una herramienta de software diseñada para realizar copias de seguridad y restauración de datos de manera fácil y eficiente. Permite crear backups automáticos de archivos, carpetas, bases de datos, correos electrónicos y sistemas completos, ya sea en dispositivos locales, redes o en la nube. HandyBackup ofrece una interfaz amigable y opciones de programación personalizables, permitiendo a los usuarios proteger sus datos contra pérdidas accidentales o fallos del sistema, al tiempo que garantiza la integridad de la información con opciones de cifrado y compresión. Es útil tanto para usuarios domésticos como para entornos empresariales que necesitan asegurar la continuidad de sus datos. (Se puede usar en S.O como Windows o Linux.)

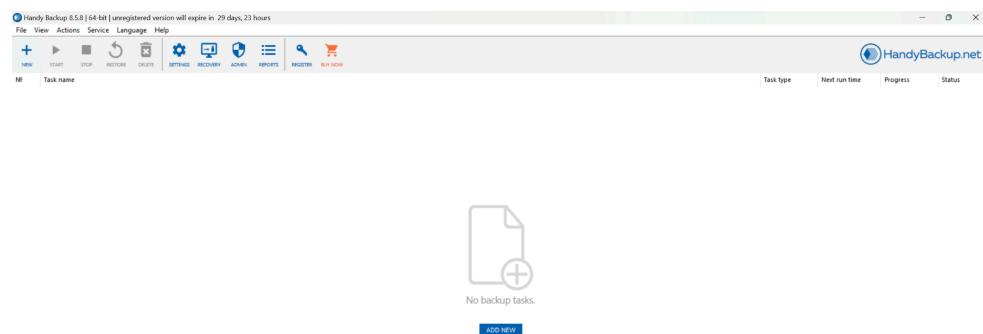
1º. Lo que haré será instalar esta herramienta en mi propia anfitriona debido a que almacenará el contenido de la base de datos de Cassandra en un backup local.



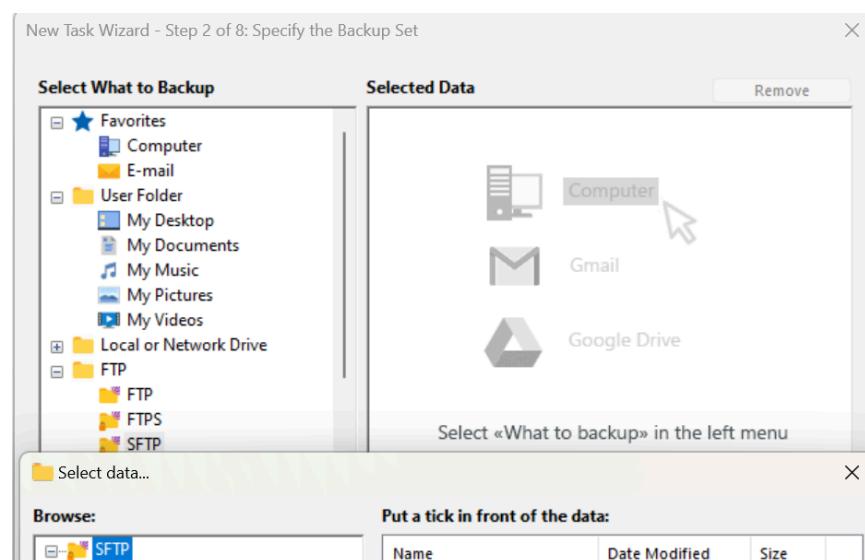
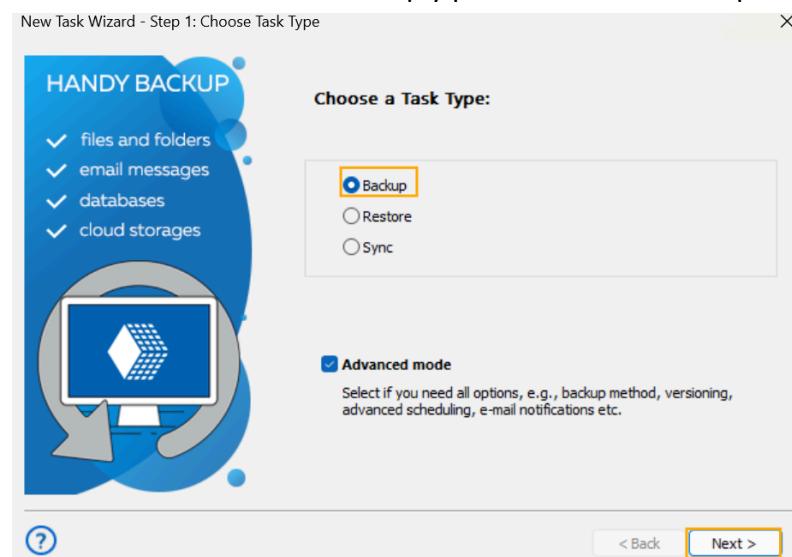
2º. Instalamos la herramienta

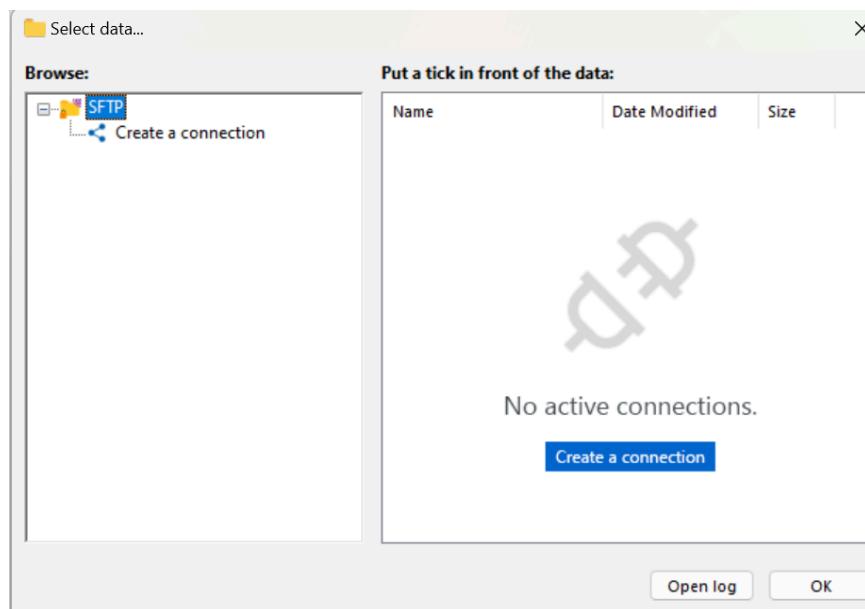


3º. Accedemos a esta herramienta y se verá un panel como el siguiente.

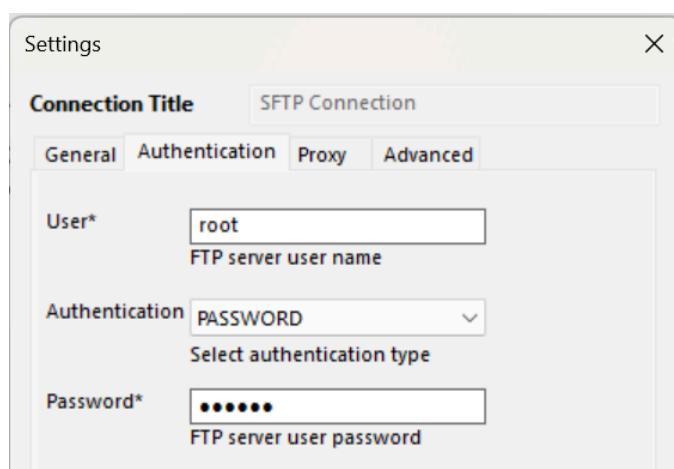
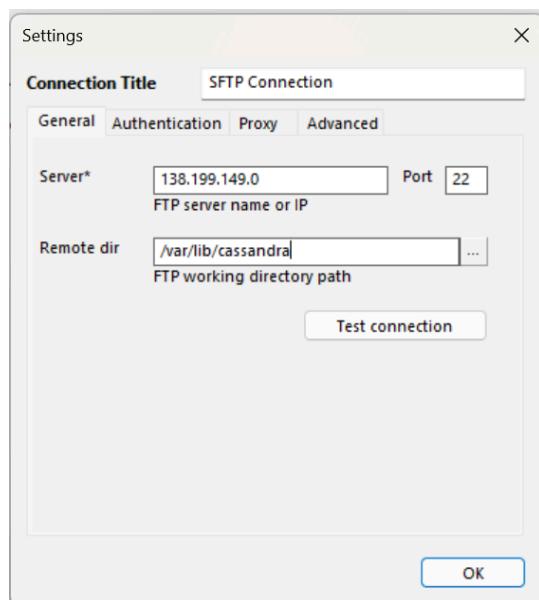


4º. Crearemos un nuevo backup y para ello usaremos el protocolo **sftp**

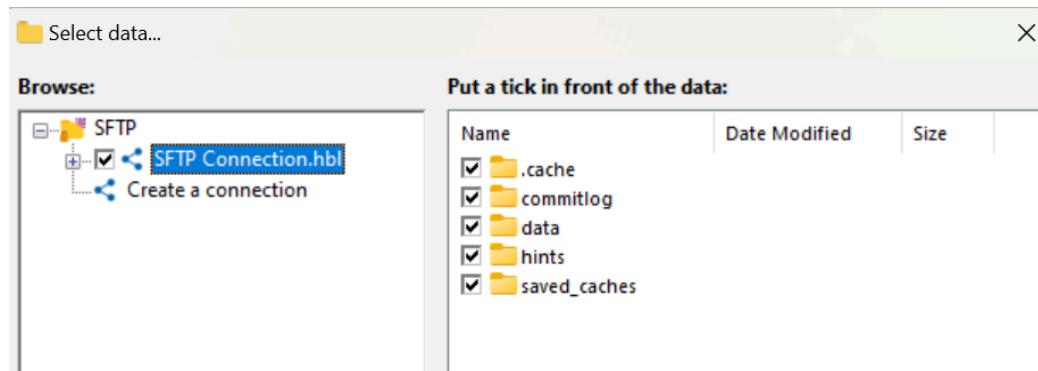




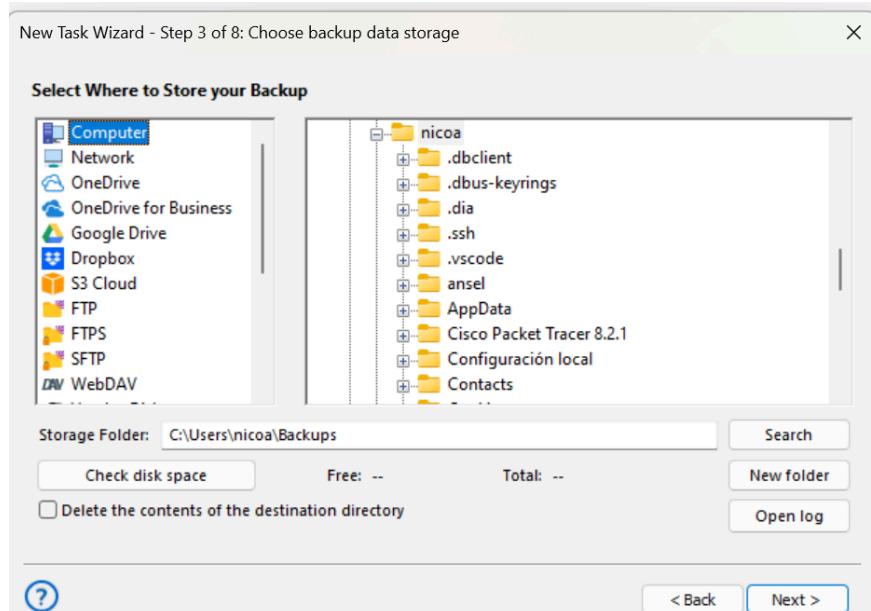
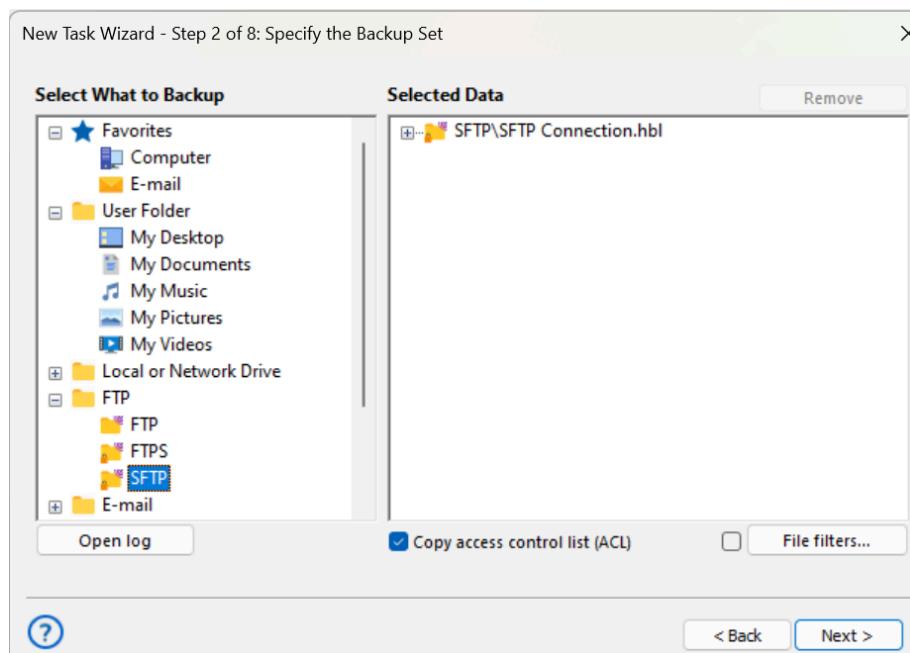
5º. Metemos la ip del vps y el directorio donde Cassandra almacena los datos.



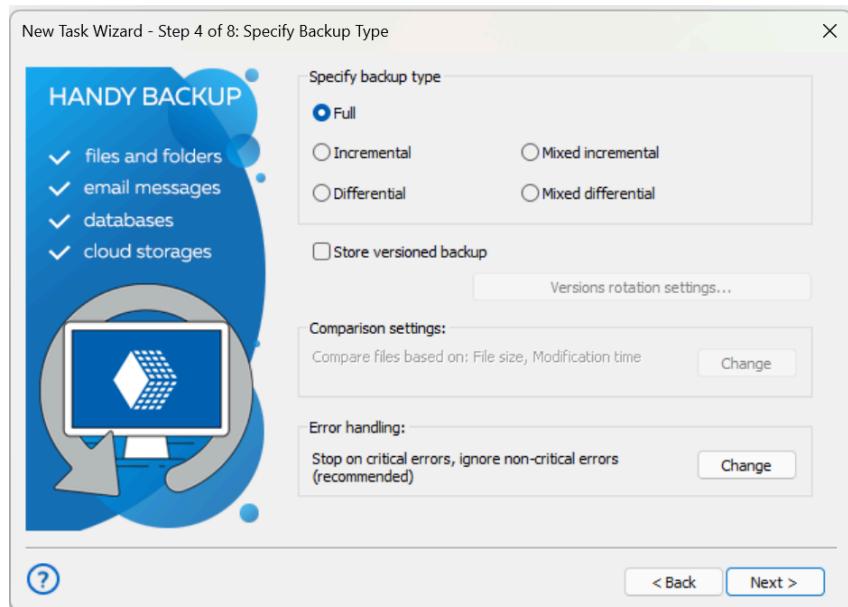
6º. Seleccionamos los archivos que queremos hacer el backup.



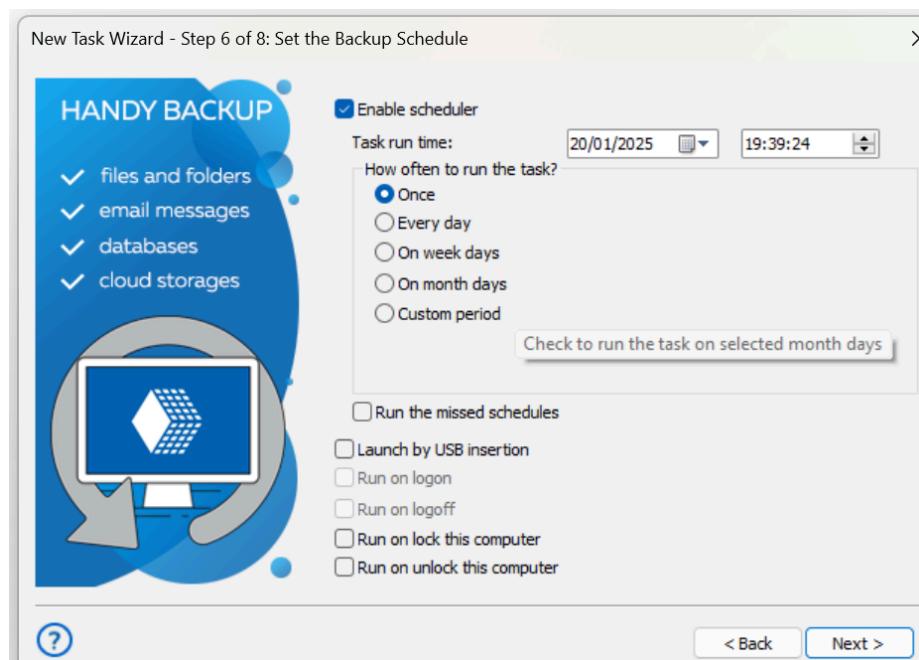
7º. Le damos a next y elegimos el directorio local para guardar el backup



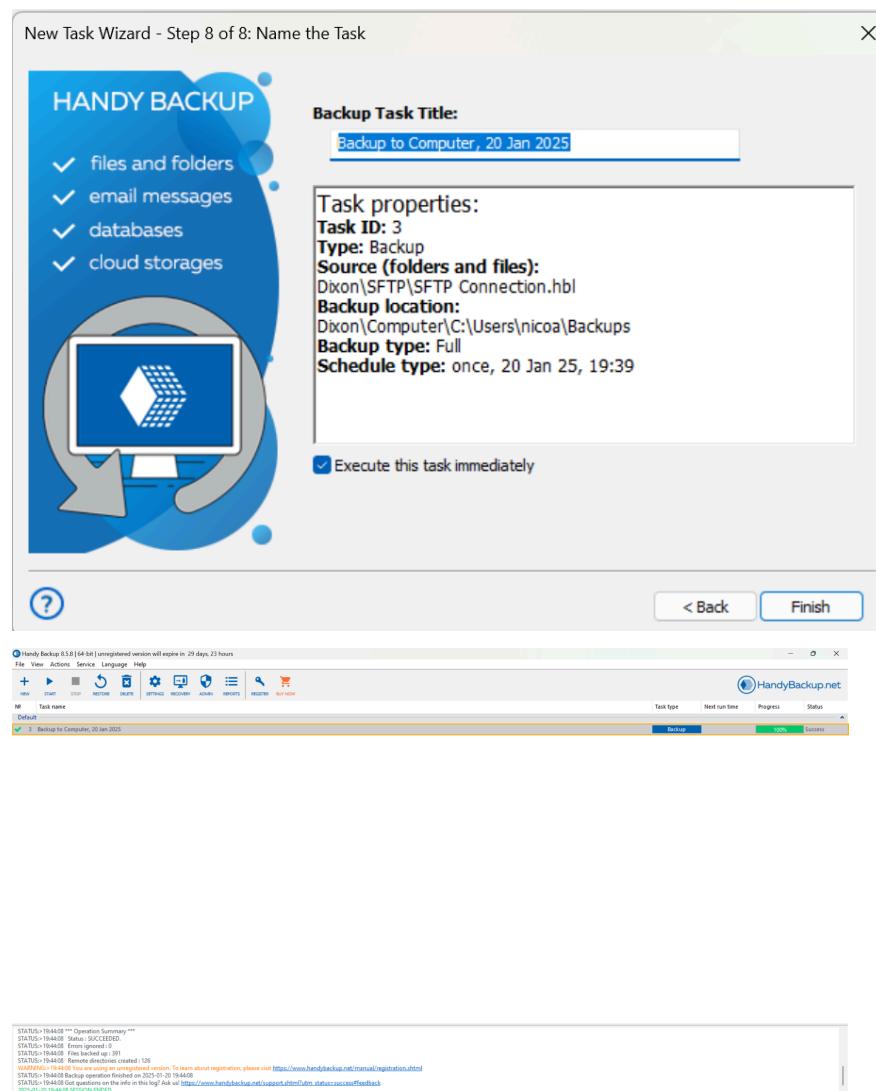
8º. Elegiremos que tipo de backup queremos (en mi caso será full)



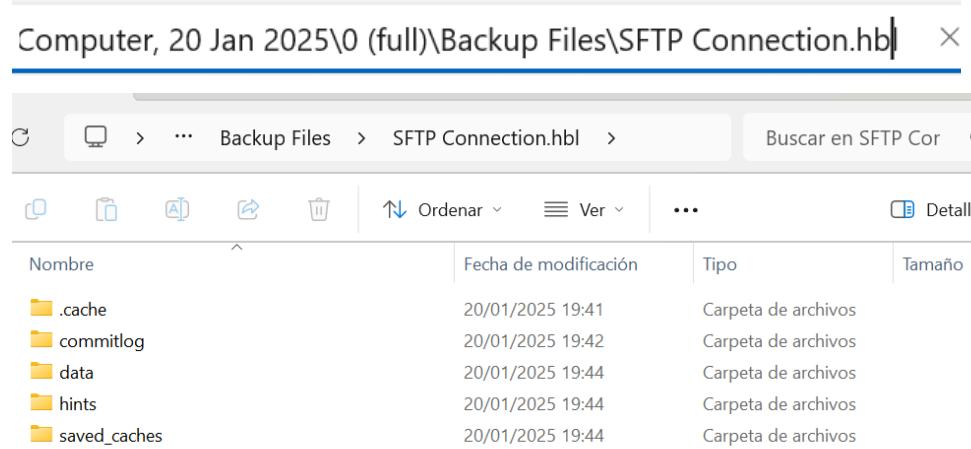
9º. Podemos programar también una fecha para hacer los backups (en mi caso será una vez y a la hora que pone)



10º. Elegimos un nombre y le damos a finish



11º. Una vez creado lo comprobaremos



Antes de restaurar eliminaremos el contenido

```
root@Tifa:~# rm -r /var/lib/cassandra/
root@Tifa:~#
root@Tifa:~# ls /var/lib/cassandra/commitlog/
ls: cannot access '/var/lib/cassandra/commitlog/': No such file or directory
root@Tifa:~# ls /var/lib/cassandra
root@Tifa:~# ls /var/lib/cassandra
ls: cannot access '/var/lib/cassandra': No such file or directory
root@Tifa:~#
```

Y para restaurar clicamos en el backup y le damos a **restore**



WARNING:> 20:01:31 SeRestorePrivilege cannot be enabled. DACL, Owner and Primary STATUS:> 20:01:31 Calculating data size... STATUS:> 20:01:31 About to copy 391 files with approximate total size of 64.32 MB STATUS:> 20:01:31 Processing root directory "SFTP Connection.hbl"...



Esperamos un rato y se habrá restaurado



```
STATUS:> 20:05:24 *** Operation Summary ***
STATUS:> 20:05:24 Status : SUCCEEDED.
STATUS:> 20:05:24 Local directories created : 0
STATUS:> 20:05:24 Files restored : 391
STATUS:> 20:05:24 Local directories created : 0
WARNING:> 20:05:24 You are using an unregistered version. To learn about registration, please visit https://www.handybackup.net/manual/registration.shtml
STATUS:> 20:05:24 Restore operation finished on 2023-01-20 20:05:24
STATUS:> 20:05:24 Get questions on the info in this log? Ask us! https://www.handybackup.net/support.shtml?utm\_status=success#feedback
2023-01-20 20:05:24 SESSION ENDED
```

Lo comprobaremos desde el VPS.

```
root@Tifa:~# ls /var/lib/cassandra/
commitlog  data  hints  saved_caches
root@Tifa:~# ls /var/lib/cassandra/
commitlog  data  hints  saved_caches
root@Tifa:~# ls /var/lib/cassandra/commitlog/
CommitLog-7-1737377919565.log  CommitLog-7-1737377919566.log
root@Tifa:~#
```

HERRAMIENTA DE SINCRONIZACIÓN DE DIRECTORIOS

Para este apartado usaré **Syncthing** que aunque es una herramienta de sincronización me permite almacenar los datos de un lugar a otro pudiendo actuar de “backup” en cierto modo.

Syncthing

Syncthing es una herramienta de sincronización de archivos de código abierto que permite la transferencia de archivos entre dispositivos de manera segura y privada, sin necesidad de un servidor central. Utiliza cifrado de extremo a extremo para garantizar que los archivos solo sean accesibles por los dispositivos que participan en la sincronización. Esto lo convierte en una excelente opción para aquellos que buscan mantener sus datos sincronizados entre dispositivos en la misma red o a través de internet sin depender de servicios en la nube.

Si bien no es un sistema de backup tradicional, Syncthing **puede funcionar como una solución de respaldo en cierta medida, ya que garantiza que los archivos se mantengan sincronizados y seguros entre varios dispositivos, proporcionando una copia actualizada de tus datos en todos los nodos participantes.**

Antes de explicar los pasos que he realizado voy a explicar lo que he querido hacer, he querido mediante 2 nodos con la herramienta **syncthing** sincronizar una carpeta de un nodo con el contenido de los datos de **cassandra** con otra en el otro nodo sin ningún dato para que actúe como “**backup**” que aunque no es lo más óptimo si desincronizas y almacenas los datos en otra carpeta si podría llegar a actuar como tal **aunque su funcionalidad como tal es**

la de sincronizar directorios pudiendo así almacenar los datos en otro directorio por si en caso de borrado de datos tener una forma de poder obtener de nuevo esos datos, aunque hay que tener en cuenta el tiempo de sincronización que se le asigna.

PASOS A SEGUIR

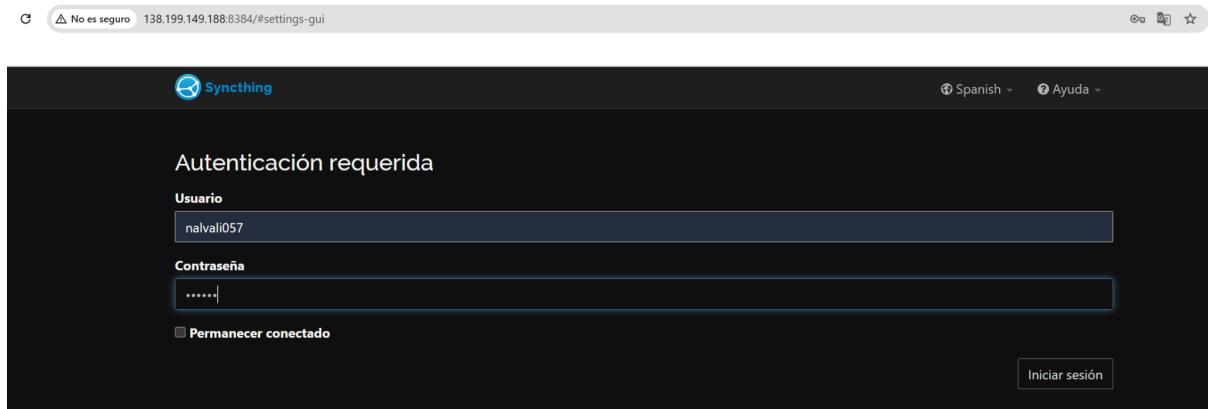
1º. Creamos un docker-compose el cuál nos permitirá levantar la herramienta

```
GNU nano 7.2                                            docker-compose.yml
#version: '3.8'
services:
  syncthing:
    image: linuxserver/syncthing
    container_name: syncthing
    environment:
      - PUID=1000 # Cambia por el ID del usuario que ejecutará el contenedor
      - PGID=1000 # Cambia por el grupo del usuario
      - TZ=Europe/Madrid # Cambia a tu zona horaria
    volumes:
      - ./config:/config # Donde Syncthing guarda su configuración
      - ./data:/data # Carpeta sincronizada
      - /var/lib/cassandra/data:/data/cassandra # Carpeta de datos de Cassandra
      - /var/lib/cassandra/commitlog:/data/commitlog # Opcional: logs de commit
    ports:
      - 8384:8384 # Puerto de la interfaz web de Syncthing
      - 22000:22000 # Puerto para sincronización
      - 21027:21027/udp # Puerto de descubrimiento local
    restart: unless-stopped
```

2º. Levantamos la herramienta

```
root@EzioCloud:~/stack/bakcups# sudo docker-compose up -d
[+] Running 10/10
  ✓ syncthing Pulled                               4.8s
    ✓ 1be6bd916534 Pull complete                  1.1s
    ✓ e1cde46db0e1 Pull complete                  1.1s
    ✓ 8fdf73aae348 Pull complete                  1.1s
    ✓ 2e51f070374a Pull complete                  1.2s
    ✓ 567188d570cb Pull complete                  2.1s
    ✓ ac04db7e683f Pull complete                  2.1s
    ✓ 2990cda5aad3 Pull complete                  2.2s
    ✓ 7b5375e55227 Pull complete                  2.9s
    ✓ 02bee8f5ef1a Pull complete                  2.9s
[+] Running 2/2
  ✓ Network bakcups_default Created             0.2s
  ✓ Container syncthing Started                 0.6s
root@EzioCloud:~/stack/bakcups#
```

3º. Accedemos a esta y le asignamos un usuario y contraseña para que sea seguro su acceso



4º. Levanto el segundo **syncthing** en otro VPS distinto.

```
GNU nano 7.2              docker-compose.yaml

services:
  syncthing:
    image: linuxserver/syncthing
    container_name: syncthing
    environment:
      - PUID=1000 # Cambia por el ID del usuario que ejecutará el contenido
      - PGID=1000 # Cambia por el grupo del usuario
      - TZ=Europe/Madrid # Cambia a tu zona horaria
    volumes:
      - ./config:/config # Donde Syncthing guarda su configuración
      - ./data:/data # Carpeta sincronizada
      - /root/stack/backup/cassandra:/backup/cassandra
    ports:
      - 8384:8384 # Puerto de la interfaz web de Syncthing
      - 22000:22000 # Puerto para sincronización
      - 21027:21027/udp # Puerto de descubrimiento local
    restart: unless-stopped
```

```
root@Tifa:~/stack/backup# docker-compose up -d
[+] Running 10/10
  ✓ syncthing Pulled          4
    ✓ 1be6bd916534 Pull complete  0
    ✓ e1cde46db0e1 Pull complete  0
    ✓ 8fdf73aae348 Pull complete  0
    ✓ 2e51f070374a Pull complete  0
    ✓ 567188d570cb Pull complete  0
    ✓ ac04db7e683f Pull complete  0
    ✓ 2990cda5aad3 Pull complete  1
    ✓ 7b5375e55227 Pull complete  1
    ✓ 02bee8f5ef1a Pull complete  1
[+] Running 2/2
  ✓ Network backup_default Created  0
  ✓ Container syncthing Started   0
root@Tifa:~/stack/backup#
```

5º. Accedo a las herramientas para verificar su funcionamiento

Carpetas

Este Dispositivo

37fe179fbe86

- Velocidad de descarga: 0 B/s (0 B)
- Velocidad de subida: 0 B/s (0 B)
- Estado Local (Total): 0 B/s (0 B)
- Oyentes: 3/3
- Descubrimiento: 4/5
- Tiempo de funcionamiento: 1m
- Identificación: SYZPTRQ
- Versión: v1.29.2, Linux (64-bit Intel/AMD Container)

Otros dispositivos

Pausar todo | Cambios recientes | + Agregar dispositivo remoto

Carpetas

Este Dispositivo

4fc712d9fbab

- Velocidad de descarga: 0 B/s (0 B)
- Velocidad de subida: 0 B/s (0 B)
- Estado Local (Total): 0 B/s (0 B)
- Oyentes: 3/3
- Descubrimiento: 4/5
- Tiempo de funcionamiento: 11m
- Identificación: EYSDPXX
- Versión: v1.29.2, Linux (64-bit Intel/AMD Container)

Otros dispositivos

Pausar todo | Cambios recientes | + Agregar dispositivo remoto

6º. Agrego un dispositivo remoto a el otro, para ello copio el id y agrego el nodo

Identificación del Dispositivo - 37fe179fbe86

SYZPTRQ-JCQ2XGR-4MYGKQP-H5IQW7-PYOYTFI-D2C72XN-RVMXDJ4-7YONIAI

Copiar
Compartir por correo electrónico
Compartir por SMS

0 B/s (0 B)
0 B/s (0 B)
0 B/s (0 B)

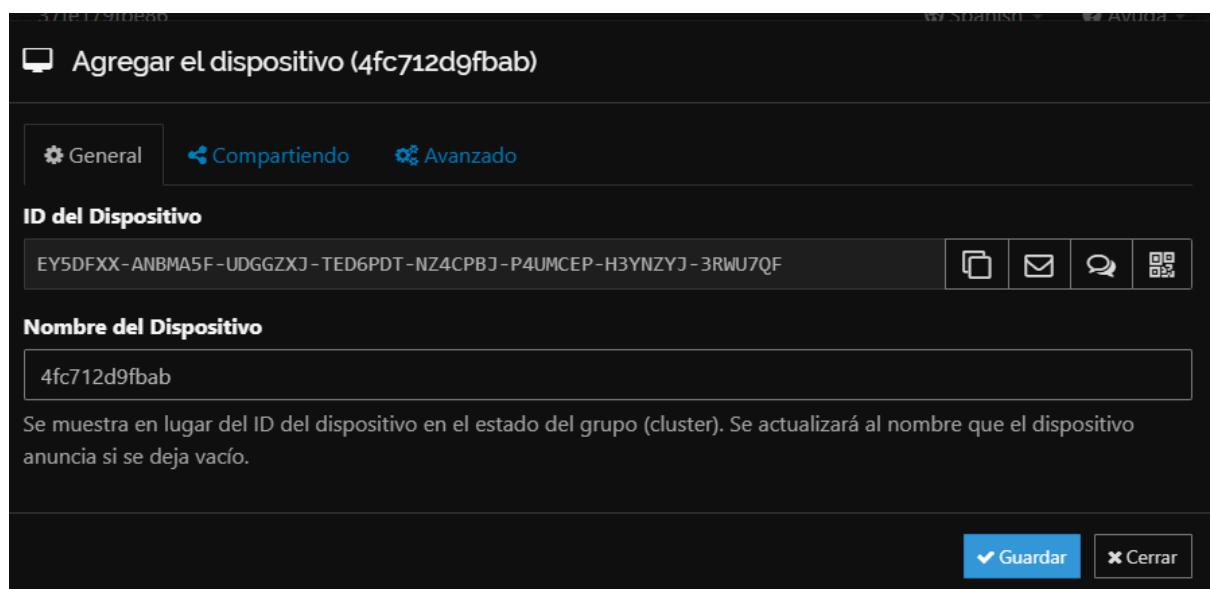
3/3
4/5
2m
SYZPTRQ
AMD Container

dispositivo remoto

Cerrar



7º. Acepto el agregar el nodo en el otro lado



De esta manera nos debería de mostrar lo siguiente

Este Dispositivo

37fe179fbe86	
Velocidad de descarga	0 B/s (6 B)
Velocidad de subida	0 B/s (6 B)
Estado Local (Total)	0 0 ~0 B
Oyentes	3/3
Descubrimiento	4/5
Tiempo de funcionamiento	4m
Identificación	5YZPTRQ
Versión	v1.29.2, Linux (64-bit Intel/AMD Container)

Otros dispositivos

4fc712d9fbab	Conectado (Sin Uso)
<input type="button" value="■■ Pausar todo"/> <input type="button" value="● Cambios recientes"/> <input type="button" value="✚ Añadir un dispositivo remoto"/>	

8º. El siguiente paso es añadir la carpeta de los datos de Cassandra en el que tiene la carpeta compartida de **Cassandra**.

Agregar Carpeta (xgyxe-t3fub)

General Compartiendo Versionado de ficheros

▼ Patrones a ignorar Avanzado

Etiqueta de la Carpeta

Cassandra Datos

Etiqueta descriptiva opcional para la carpeta. Puede ser diferente en cada dispositivo.

ID de carpeta

xgyxe-t3fub

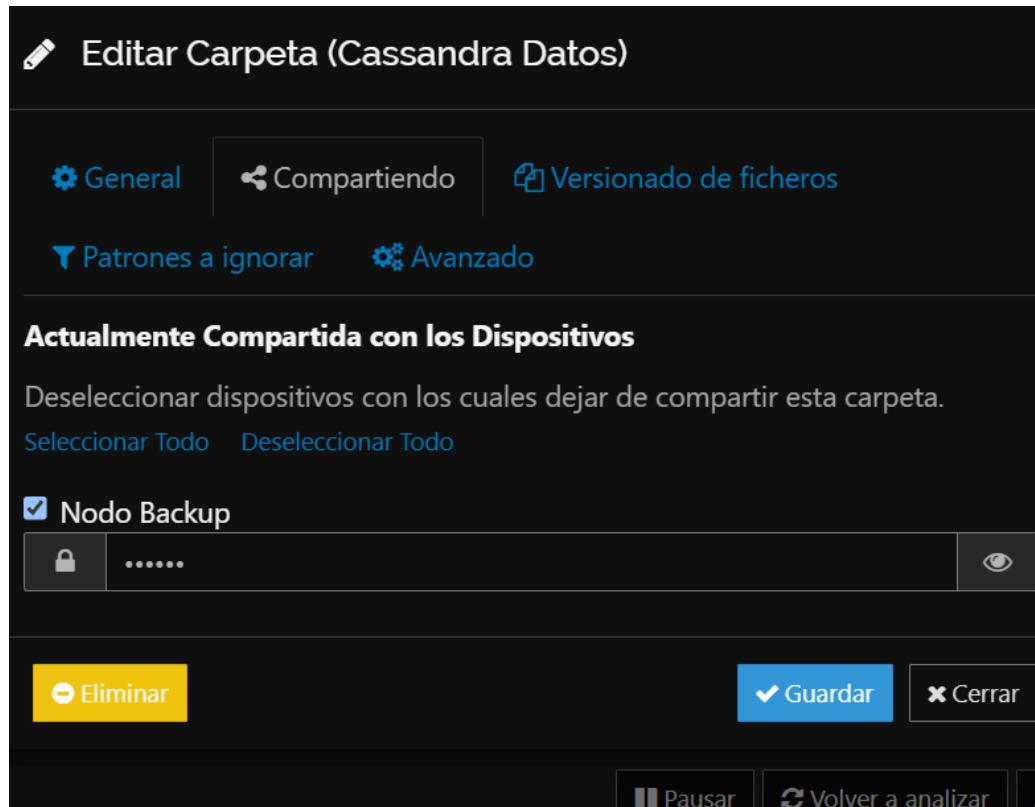
Identificador requerido para la carpeta. Debe ser el mismo en todos los dispositivos del clúster. Cuando añada una nueva carpeta, tenga en cuenta que su ID se usa para unir carpetas entre dispositivos. Son sensibles a las mayúsculas y deben coincidir exactamente entre todos los dispositivos.

Ruta de la carpeta

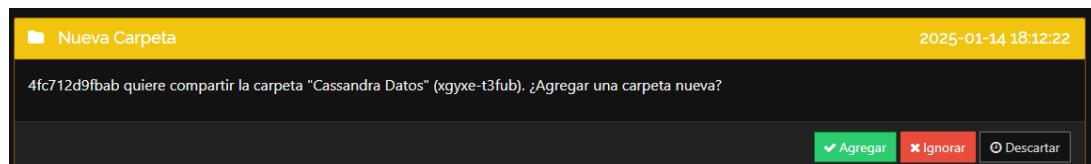
/data/cassandra

Ruta a la carpeta en el dispositivo local. Se creará la carpeta si no existe. El carácter de la tilde (~) se puede utilizar como abreviatura de `/config`.

Y añadimos el nodo a compartir



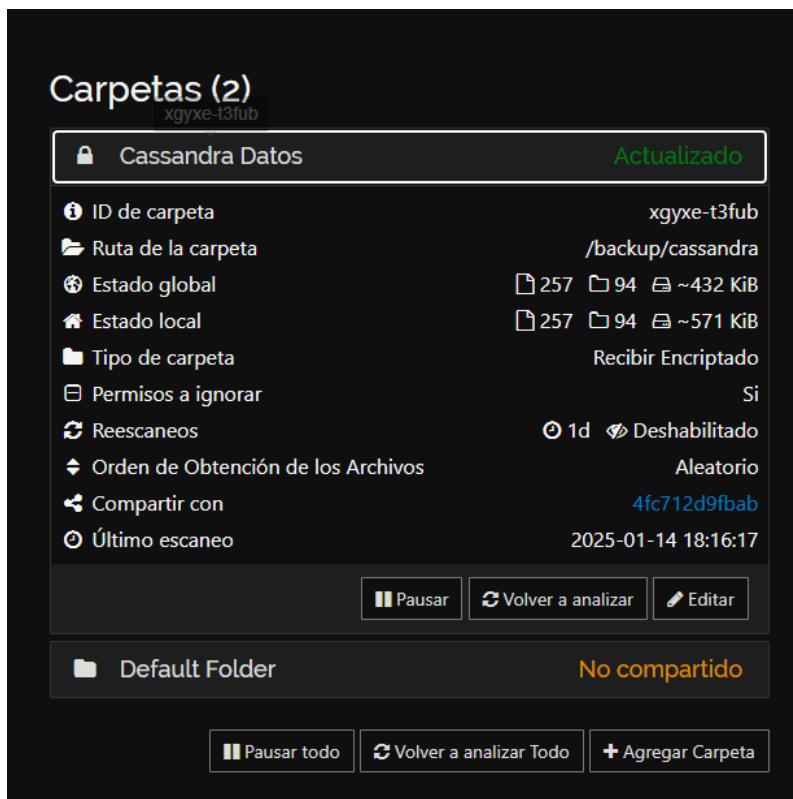
Luego aceptamos el agregar una nueva carpeta y añadiremos la de backups



Antes de agregar le damos los permisos

```
root@Tifa:~/stack/backup# sudo chown -R 1000:1000 /root/stack/backup/cassandra/
```

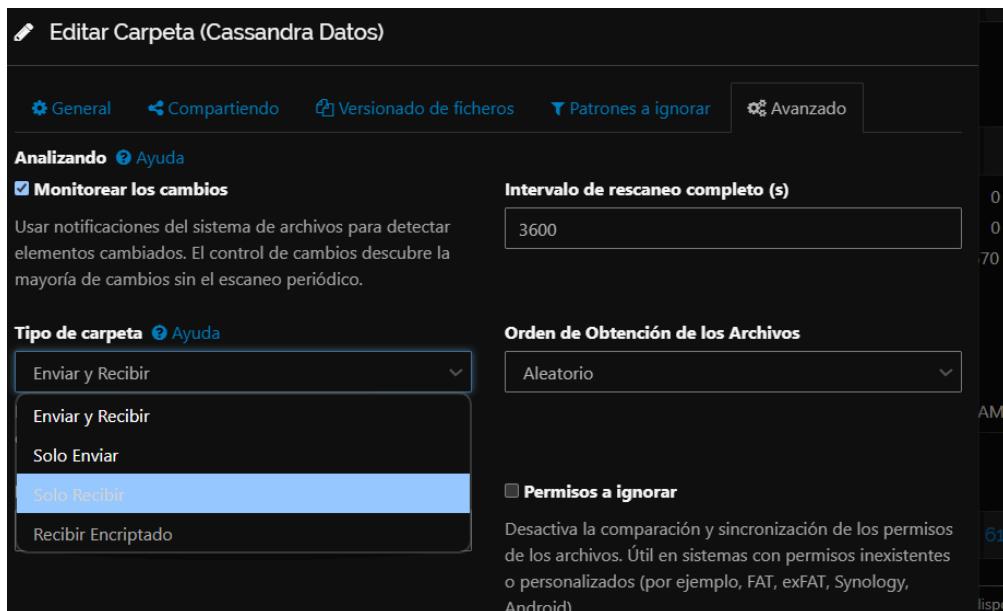
9º. Aceptamos para agregar los datos



10º. De esta manera ya estaría el “**backup**” realizado, aunque realmente son 2 directorios sincronizados (como se puede apreciar aparece datos de forma cifrada o sin cifrar, esto se debe a que le he dado en un primer momento a cifrado y luego le he dado a sin cifrar)

```
root@Tifa:~/stack/backup# ls cassandra/
0.syncthing-enc/           8.syncthing-enc/           G.syncthing-enc/           O.syncthing-enc/           system.auth/
1.syncthing-enc/           9.syncthing-enc/           H.syncthing-enc/           P.syncthing-enc/           system.distributed/
2.syncthing-enc/           A.syncthing-enc/           I.syncthing-enc/           Q.syncthing-enc/           system.schema/
3.syncthing-enc/           B.syncthing-enc/           J.syncthing-enc/           R.syncthing-enc/           system.traces/
4.syncthing-enc/           C.syncthing-enc/           K.syncthing-enc/           S.syncthing-enc/           T.syncthing-enc/
5.syncthing-enc/           D.syncthing-enc/           L.syncthing-enc/           .stfolder/               U.syncthing-enc/
6.syncthing-enc/           E.syncthing-enc/           M.syncthing-enc/           .stfolder.removed-20250114-182150/ V.syncthing-enc/
7.syncthing-enc/           F.syncthing-enc/           N.syncthing-enc/           system/
root@Tifa:~/stack/backup# ls cassandra/.stfolder
syncthing-folder-3a031c.txt
root@Tifa:~/stack/backup# ls cassandra/system
system/          system.auth/    system_distributed/   system_schema/
available_ranges_c539fcabd65a1d18133d25605643ee3/   paxos_repair_history-ecb8666740b23316bb91e612c8047457/   size_estimates-618f817b005f3678b8a453f3930b8e86/
available_ranges_v2-4224a0882ac93d0c889dfbb5f0facda0/   _paxos_repair_state/   sstable_activity-5a1ff267ace03f128563cfae6103c65e/
batches-919a4bc57a33573bb3e13fc3ff68465/   peer_events-59dfeaea8db2334191ef00974d81484/   sstable_activity_v2-62eff31f3be8310c8d298963439c1288/
builtInView-403c5a8e6a3d7691016b0c938494a/   _paxos_repair_state/   table_estimates-1/6c39cd093d3a5a2189e8005a5676be/
compaction_history-b4dbcb7bd4c493fb5b8fbce66434832ca/   peer_events_v2-0eb50657e40138e0950793219ae8011/   top_partitions-7e5a361c317c351fb15fffd8af3dd4b/
IndexerInfo-9f5c5a9932939a50804331163/   peers-37f71aca7dc2383ba70672528a04d4ff/   transferred_ranges-6cad20f7d4f53af2b6e20da3c6c1f83/
loc-7ad54392bddd35a68417a0416609377/   peers_v2-c4925fbbe595babfb070f9250ed818e/   transferred_ranges_v2-1ff781la/d13aa2a9986f4932270af5/
paxos_b7bf70cfd0a34108c052eef6bb7c2d/   prepared_statements-18a92c576a0c3841ba718cd529849fef/   view_builds_in_progress-6c22dff66c3bd3df6b74d21179c6a9fe9/
root@Tifa:~/stack/backup# ls cassandra/system.auth/
network_permissions-d46780c22fc13d694c158d9fb0c23/   role_permissions-3afbe79f219431a7add7f5ab90d8ec9c/
resource_role_permissions_index-f52f2fbad91f13946bd25d5da3a5c35ec/   roles-5bc52802de2535edaeb180eeceb090/
role_members-0ecdaa8778fb53e6088d174fb3f5e50d/
```

La opción de cifrado se puede establecer en este apartado (para ello cuando compartes la carpeta es necesario establecer una contraseña, de esta forma lo enviará cifrado; si tu lo envias sin contraseña lo enviará sin cifrar)



Hay que tener en cuenta que al ser una herramienta de sincronización si haces algún cambio se pueden llegar a perder los datos, por lo que una vez sincronizados los datos lo suyo es parar la sincronización y pasar los archivos a otra carpeta o programar la sincronización o ejecutarla en momentos en los que te sea necesario para respaldar datos si es que la deseas como herramienta de backup.

MONITORIZACIÓN & LOGS

PARÁMETROS

Cqlsh

- **Cassandra_Process:** Muestra información del estado del proceso de Cassandra.
- **Cassandra_CPU_Usage:** Muestra el uso de la cpu por el proceso de Cassandra en %.
- **Cassandra_Memory_Use:** Muestra el uso de memoria por el proceso de Cassandra en %.
- **Cassandra_Error_Log_Messages:** Muestra la cantidad de mensajes de error en el log de Cassandra.
- **Cassandra_Warning_Log_Messages:** Muestra la cantidad de mensajes de advertencia en el log de Cassandra.
- **Cassandra_Network_Connections:** Muestra la cantidad de conexiones de clientes.
- **Cassandra_Thrift_Server_Status:** Muestra el estado del servidor Thrift.
- **Cassandra_Cluster_Status :**Muestra el estado del clúster principal.
- **Cassandra_Key_Cache_Size:** Muestra el tamaño de la “key cache” en kbs.
- **Cassandra_Active_Commands:** Muestra la cantidad de comandos activos o tareas en ejecución.
- **Cassandra_Pending_Commands:** Muestra la cantidad de comandos pendientes.
- **Cassandra_Completed_Commands:** Muestra la cantidad de comandos completados.
- **Cassandra_Active_Responses:** Muestra la cantidad de respuestas activas a comandos.
- **Cassandra_Pending_Responses:** Muestra la cantidad de respuestas pendientes a comandos.
- **Cassandra_Completed_Responses:** Muestra la cantidad de respuestas completas a comandos.
- **Cassandra_Nodetool_Configuration:** Detecta si Nodetool está funcionando correctamente.

Logs importantes

- **system.log:** Principal log del sistema que registra la mayoría de las actividades del nodo, como operaciones de escritura/lectura, eventos de compacción, y errores generales.
- **debug.log:** Utilizado para mensajes detallados y depuración (si está habilitado).
- **gc.log:** Registro de actividades de Garbage Collection (GC), que puede ser útil para identificar problemas de rendimiento relacionados con el manejo de memoria.
- **audit.log:** Contiene registros de auditoría si se habilita la auditoría en Cassandra.

Parámetros Monitoreo en Tiempo Real

- **nodetool status:** Ver el estado de los nodos del clúster.
- **nodetool tpstats:** Monitorear las estadísticas de los "Thread Pools" en tiempo real.
- **nodetool compactionstats:** Ver el estado actual de las compactaciones en el nodo.
- **nodetool netstats:** Mostrar estadísticas de red del nodo, incluidas las transmisiones en curso y recibidas.
- **nodetool cfstats:** Ver estadísticas detalladas de las tablas.
- **nodetool repair:** Iniciar un proceso de reparación para garantizar la consistencia de datos entre réplicas.
- **nodetool flush:** Forzar la escritura de datos en memoria (memtables) al disco.
- **nodetool describecluster:** Mostrar información básica del clúster, como el nombre, UUID y versión del esquema.
- **nodetool snapshot:** Crear una instantánea (snapshot) de todas las tablas en un nodo.
- **nodetool decommission:** Sacar un nodo del clúster de manera ordenada.

Rutas de logs

Tipo de Log	Ruta Predeterminada	Descripción
-------------	---------------------	-------------

Log Principal (system.log)	/var/log/cassandra/system.log	Registra eventos generales del nodo, como inicio, errores, cambios en el clúster, etc.
Log de Depuración (debug.log)	/var/log/cassandra/debug.log	Incluye mensajes detallados para depuración si se habilita el nivel DEBUG.
Log de Recolección de Basura (gc.log)	/var/log/cassandra/gc.log	Registra eventos relacionados con Garbage Collection (GC) y gestión de memoria.
Log de Auditoría (audit.log)	/var/log/cassandra/audit.log	Registra eventos sensibles como operaciones de usuario (solo si la auditoría está habilitada).
Log de Inicio (startup.log)	/var/log/cassandra/startup.log	Captura mensajes generados al iniciar el proceso de Cassandra.

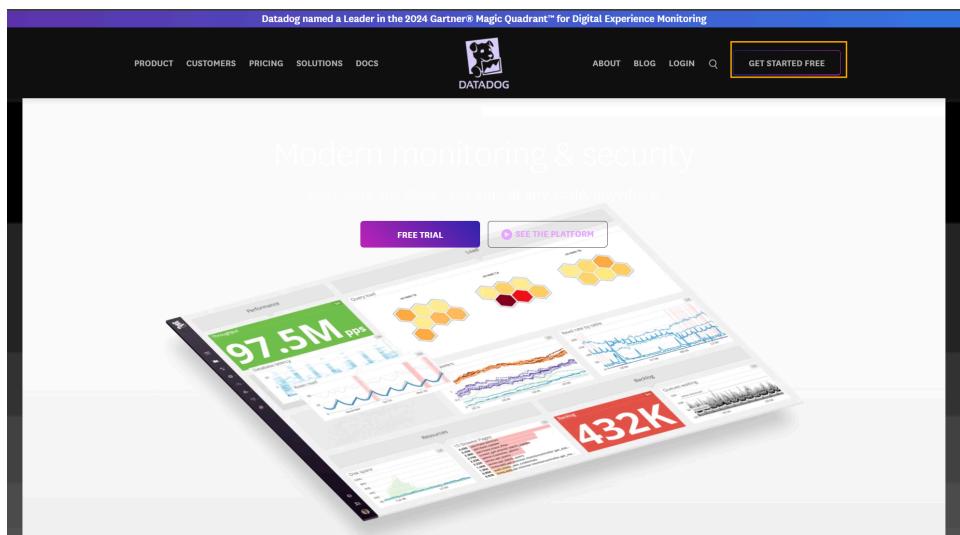
HERRAMIENTA DE MONITORIZACIÓN (DATADOG)

Datadog es una herramienta de monitoreo y análisis de datos en la nube (es de pago pero tiene una prueba gratuita de 14 días). Se utiliza para observar el rendimiento de aplicaciones, infraestructura y servicios a través de paneles visuales. Con Datadog, puedes:

- **Centralizar el monitoreo:** Visualizar métricas de múltiples servidores, bases de datos, contenedores y servicios en un solo lugar.
- **Detectar problemas de rendimiento:** Configurar alertas para recibir notificaciones sobre problemas antes de que afecten a los usuarios.
- **Optimizar aplicaciones:** Conocer qué partes de una aplicación están ralentizándola y cómo mejorarlas.

PASOS PARA PODER USAR DATADOG

1º. Creamos una cuenta y obtenemos la clave API (para ello ingresaré en el modo get started free (que es una prueba de 14 días)



Get started with Datadog [Already have an account?](#)

Try it free for 14 days and monitor as many servers as you like.
*Required Fields

Region*
Please choose carefully. You can't migrate data between regions.
Where do you want your data housed?
Europe (EU1)

Business Email*
nalvali057@g.educaand.es

Full Name*
Nico

Company*
ninguna

Password*

Phone
[Input field]

*Required fields. By signing up, you agree to the [Master Subscription Agreement](#), [Privacy Policy](#) and [Cookie Policy](#)

Sign up with Google Sign up

2º. Una vez nos registremos podremos ver todas las opciones siguientes con las que podemos integrar DataDog (pero yo cogeré cassandra)

3º. Seleccionamos el sistema operativo para instalar el agente (Debian 12 que es el S.O de mi VPS donde tengo Cassandra)

4º. Ejecutamos el comando siguiente para unir el **VPS** a **DataDog**, lo que hace realmente es instalarse el agente de DataDog porque funciona con agente.

```
root@EzioCloud:~/stack/bakcups# DD_API_KEY=1eb7ff9beff5067ad733f11b49c4649 DD_SITE="datadoghq.eu" bash -c "$(curl -L https://install.datadoghq.com/scripts/install_script_agent7.sh)"
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left Speed
100 70510  100 70510     0      0 129K  0 ---:--:--:--:--:-- 129K

* Datadog Agent 7 install script v1.36.1
/usr/bin/systemctl
* Installing curl gnupg

Hit:1 http://mirror.hetzner.com/debian/packages bookworm InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Get:3 http://mirror.hetzner.com/debian/packages bookworm-updates InRelease [55.4 kB]
Hit:4 http://mirror.hetzner.com/debian/security bookworm-security InRelease
Get:5 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:6 http://security.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:7 http://deb.debian.org/debian unstable InRelease [205 kB]
Hit:8 http://download.docker.com/linux/debian bookworm InRelease
Get:9 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [241 kB]
Get:10 http://deb.debian.org/debian unstable/main amd64 Packages [10.0 MB]
Get:11 http://security.debian.org/debian-security bookworm-security/main Translation-en [142 kB]
Get:12 http://deb.debian.org/debian unstable/main Translation-en [7,323 kB]
```

```
* Setting SITE in the Datadog Agent configuration: /etc/datadog-agent/datadog.yaml

/usr/bin/systemctl
* Starting the Datadog Agent...

Your Datadog Agent is running and functioning properly.
It will continue to run in the background and submit metrics to Datadog.
If you ever want to stop the Datadog Agent, run:

    systemctl stop datadog-agent

And to run it again run:

    systemctl start datadog-agent

Consider adding dd-agent to the docker group to enable the docker support, run:

    sudo usermod -a -G docker dd-agent

root@EzioCloud:~/stack/bakcups#
```

5º. Reiniciamos el servicio (`systemctl restart datadog-agent.service`)

```
root@EzioCloud:~/stack/bakcups# sudo systemctl restart datadog-agent.service
root@EzioCloud:~/stack/bakcups# sudo systemctl status datadog-agent.service
● datadog-agent.service - Datadog Agent
   Loaded: loaded (/usr/lib/systemd/system/datadog-agent.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-01-14 18:51:27 UTC; 7s ago
     Main PID: 56999 (agent)
        Tasks: 8 (limit: 4531)
       Memory: 72.9M (peak: 75.3M)
         CPU: 1.620s
      CGroup: /system.slice/datadog-agent.service
              └─56999 /opt/datadog-agent/bin/agent run -p /opt/datadog-agent/run/agent.pid

Jan 14 18:51:28 EzioCloud agent[56999]: 2025-01-14 18:51:28 UTC | CORE | INFO | [pkg/collector/corechecks/containerimage/check.go:126 in Run] | Starting long-running check "container"
Jan 14 18:51:30 EzioCloud agent[56999]: 2025-01-14 18:51:30 UTC | CORE | INFO | [pkg/collector/corechecks/containerimage/check.go:59 in CheckFinished] | check: container image | Done running check...
Jan 14 18:51:30 EzioCloud agent[56999]: 2025-01-14 18:51:30 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:40 in CheckStarted] | check:ntp | Running check...
Jan 14 18:51:30 EzioCloud agent[56999]: 2025-01-14 18:51:30 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:40 in CheckStarted] | check:service discovery | Running check...
Jan 14 18:51:30 EzioCloud agent[56999]: 2025-01-14 18:51:30 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:59 in CheckFinished] | check:service discovery | Done running check...
Jan 14 18:51:30 EzioCloud agent[56999]: 2025-01-14 18:51:30 UTC | CORE | INFO | [pkg/utility/containers/metrics/provider/registry.go:102 in collectorDiscovery] | Container metrics provider
Jan 14 18:51:33 EzioCloud agent[56999]: 2025-01-14 18:51:33 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:40 in CheckStarted] | check:telemetry | Running check...
Jan 14 18:51:33 EzioCloud agent[56999]: 2025-01-14 18:51:33 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:59 in CheckFinished] | check:telemetry | Done running check...
Jan 14 18:51:33 EzioCloud agent[56999]: 2025-01-14 18:51:34 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:40 in CheckStarted] | check:memory | Running check...
Jan 14 18:51:34 EzioCloud agent[56999]: 2025-01-14 18:51:34 UTC | CORE | INFO | [pkg/collector/worker/check_logger.go:59 in CheckFinished] | check:memory | Done running check...
```

6º. Finalizamos la instalación en la página y accederemos al panel

Welcome, Nico!

Your hosts (so far)

Create a dashboard

Create a monitor

Increase your knowledge

Invite teammates

Install an integration

You have 0 integrations installed but 0 waiting from your stack (we remembered you mentioned these earlier). [See All 400+ Integrations](#)

Available in your stack

cassandra + Add

Cloud Providers

AWS

Azure

Nos iremos a integración y seleccionaremos a cassandra configuración

Welcome, Nico! Get Started

You have 14 days left in your trial. Upgrade

Integrations Marketplace Agent Fleet Automation NEW Reference Tables PREVIEW ...

Search integrations All Statuses

AWS Azure Google Cloud AI/ML Alerting Automation Collaboration

Configuration & Deployment Containers Data Stores Developer Tools Kubernetes

Message Queues Network OS & System Security Testing All Categories

Autodetected Integrations

Cassandra by Datadog DETECTED

Docker by Datadog DETECTED

SSH by Datadog DETECTED

.NET

.NET CLR by Datadog

.NET Runtime Metrics by Datadog

1Password by Datadog

ably

Abnormal

Abnormal Security by Datadog

Active Directory by Datadog

7º. Instalaremos la **integración** con **Cassandra** y seguiremos los pasos que se indican. (esto se debe a que primero hemos instalado el agente de **DataDog** en el VPS pero ahora hay que configurar **Cassandra** para poder obtener información acerca de esta)

The screenshot shows the Datadog interface for monitoring a Cassandra cluster. At the top, there's a logo for 'cassandra' with an eye icon. Below it, the text 'Cassandra' and 'by Datadog'. A sub-header says 'Track cluster performance, capacity, overall health, and much more.' A prominent 'DETECTED' button is visible. The navigation bar includes 'Overview', 'Configure' (which is selected), 'Data Collected', 'Hosts', 'Monitoring Resources', 'Support', and 'Release Notes'. Under 'Configuration', there's a 'Installation' section noting that the Cassandra check is included in the Datadog Agent package. It also provides instructions for collecting metrics, mentioning a limit of 350 metrics per instance and pointing to the JMX documentation for customization. The 'Metric collection' section lists two steps: activating collection via a configuration file and restarting the agent. The 'Log collection' section is available for agent versions >6.0, with instructions for Kubernetes and Docker environments. It shows a sample YAML configuration for log collection from Cassandra logs.

```

logs:
  - type: file
    path: /var/log/cassandra/*.log
    source: cassandra
    service: myapplication
    log_processing_rules:

```

8º. Configuramos la recolección de logs en datadog y cassandra (**/etc/datadog-agent/datadog.yaml**) y (**/etc/datadog-agent/conf.d/cassandra.d/conf.yaml**) para poder habilitar la monitorización de logs acerca de cassandra.

```

GNU nano 7.2          /etc/datadog-agent/datadog.yaml
## Log collection Configuration ##
#####
## @param logs_enabled - boolean - optional - default: false
## @env DD_LOGS_ENABLED - boolean - optional - default: false
## Enable Datadog Agent log collection by setting logs_enabled to true
#
logs_enabled: true

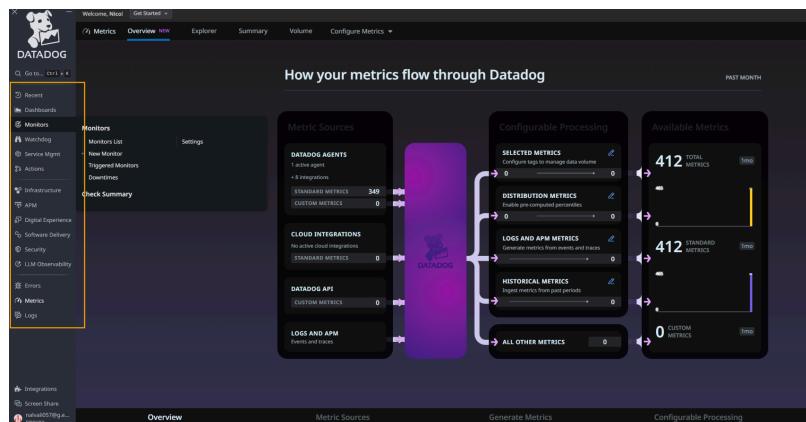
```

```
GNU nano 7.2      /etc/datadog-agent/conf.d/cassandra.d/conf.yaml
logs:
  - type: file
    path: /var/log/cassandra/*.log
    source: cassandra
    service: myapplication
    log_processing_rules:
      - type: multi_line
        name: log_start_with_date
        pattern: '[A-Z]+ +\[\[^ \]\]+\] +\d{4}-\d{2}-\d{2}+'
```

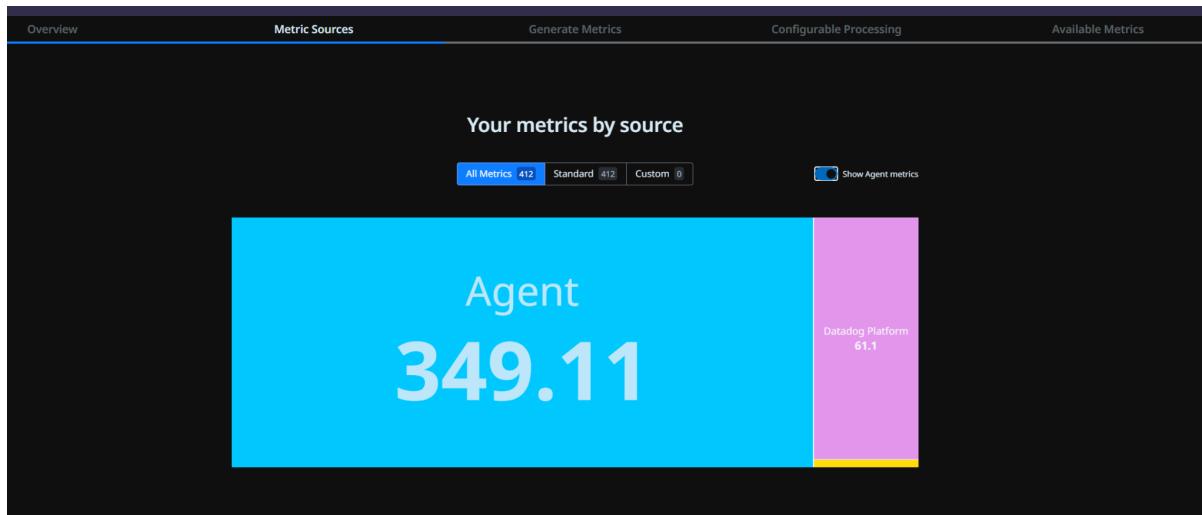
9º. Reiniciamos el agente

```
root@Tifa:~# sudo nano /etc/datadog-agent/conf.d/cassandra.d/conf.yaml
root@Tifa:~# sudo systemctl restart datadog-agent.service
root@Tifa:~# sudo systemctl status datadog-agent.service
● datadog-agent.service - Datadog Agent
    Loaded: loaded (/usr/lib/systemd/system/datadog-agent.service; enabled; p
      Active: active (running) since Tue 2025-01-14 20:23:28 UTC; 1s ago
  Invocation: ced66dae3a3248ecbd8cb38242fbf420
    Main PID: 94332 (agent)
       Tasks: 8 (limit: 4531)
      Memory: 69.5M (peak: 70M)
        CPU: 1.624s
      CGroup: /system.slice/datadog-agent.service
              └─94332 /opt/datadog-agent/bin/agent/run -p /opt/datadog-ag>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (co>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (co>
Jan 14 20:23:29 Tifa agent[94332]: 2025-01-14 20:23:29 UTC | CORE | INFO | (pk>
root@Tifa:~#
```

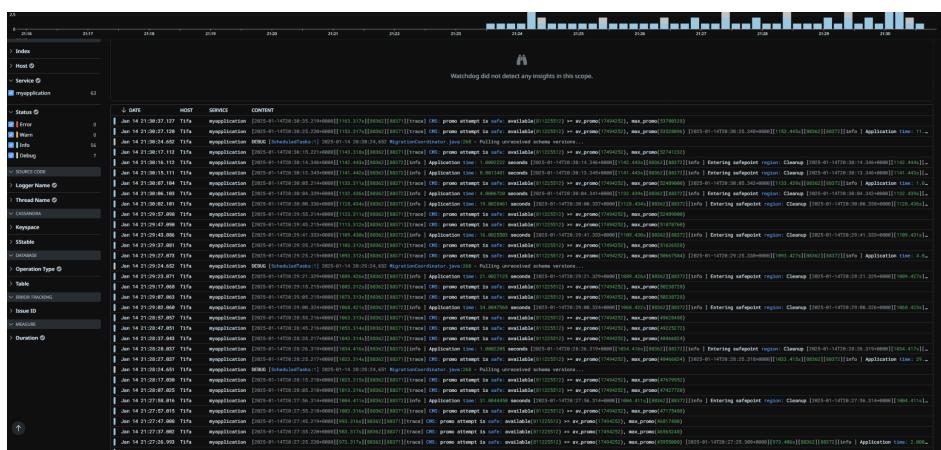
10º. Como se pueden ver hay muchos apartados los cuales se pueden usar para monitorizar cassandra



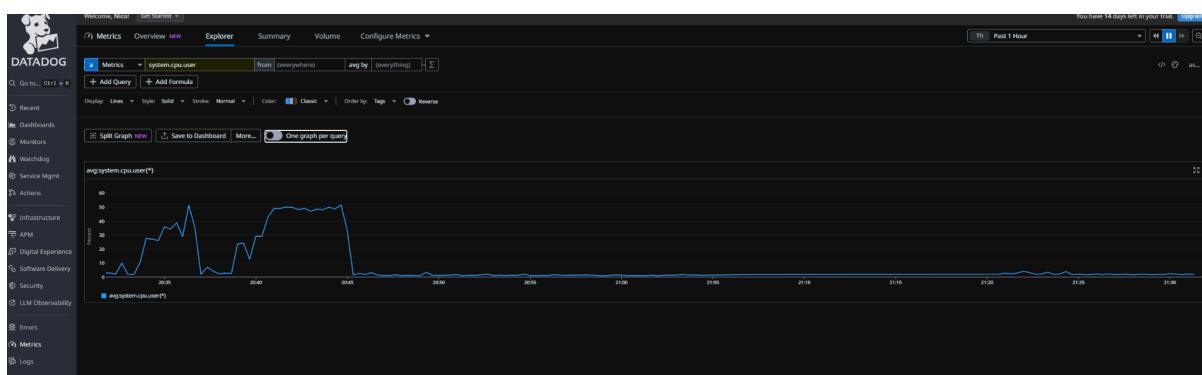
11º. En cuanto a las métricas podemos ver que mi agente tiene 349 disponibles.



12º. En este caso estoy caso estoy viendo los logs de cassandra



13º. También se puede ver la métrica de la cpu (se pueden configurar más pero en mi caso he cogido una del sistema porque sirve perfectamente para mostrar cómo de estresado está el VPS con Cassandra)

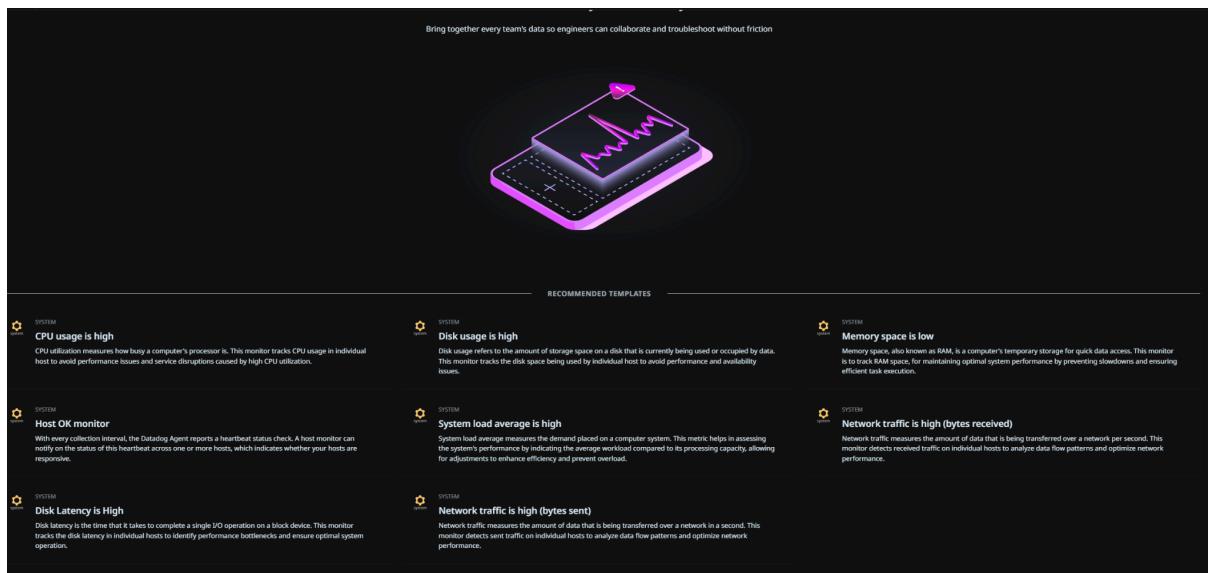


14º. También puedo ver los nodos de la infraestructura (en mi caso se ven 2 porque uní 2 veces el agente, pero hay 1 que no está configurado y no me recoge nada acerca de este)

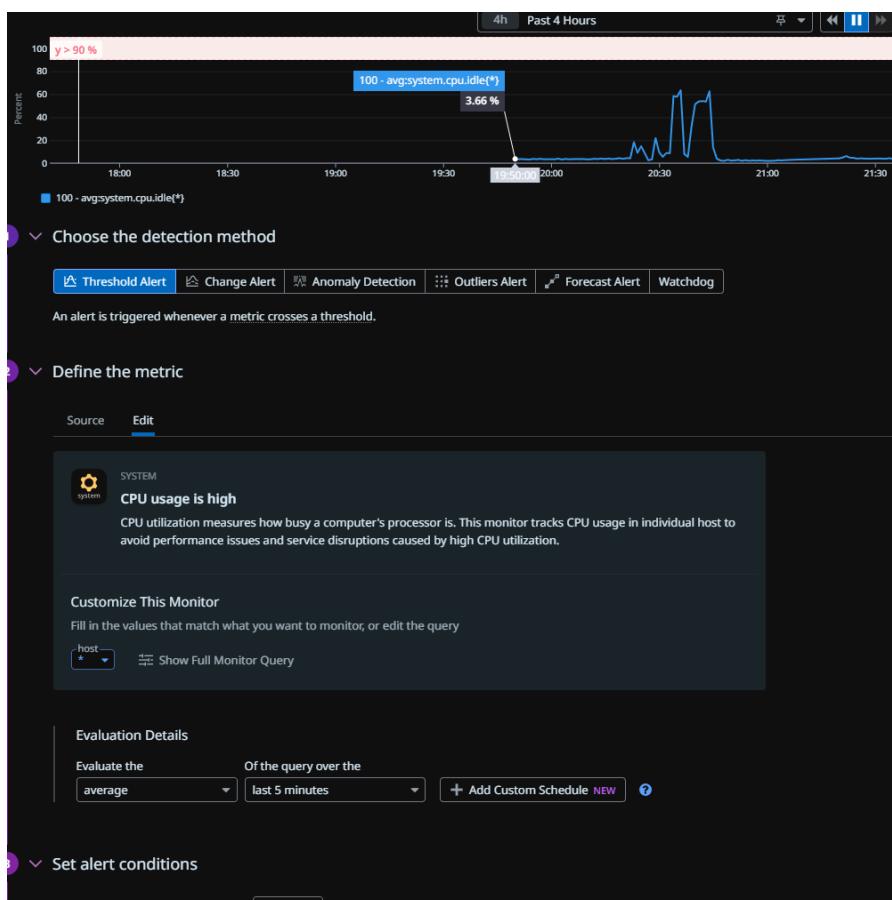
15º. Si seleccionas el nodo puedes ver información como las métricas, logs, etc ...

De esta manera es lo mismo que seleccionar los apartados de uno en uno pero de este modo te salen todos del tirón.

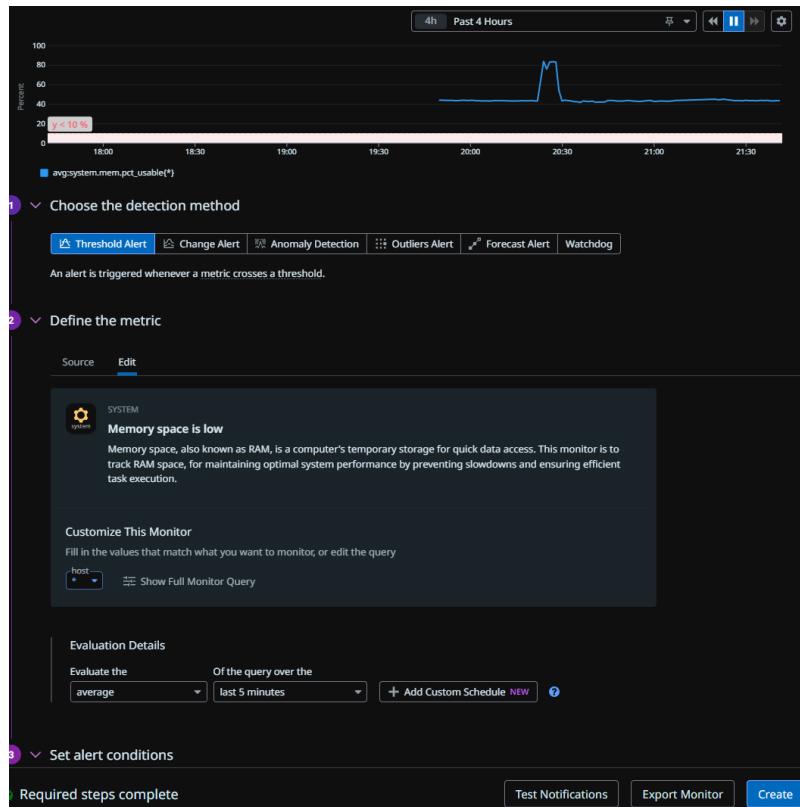
17º. También en la opción **monitors** se pueden ver las siguientes opciones (Son plantillas que se pueden usar para monitorizar recursos del sistema acerca del servidor que contiene cassandra).



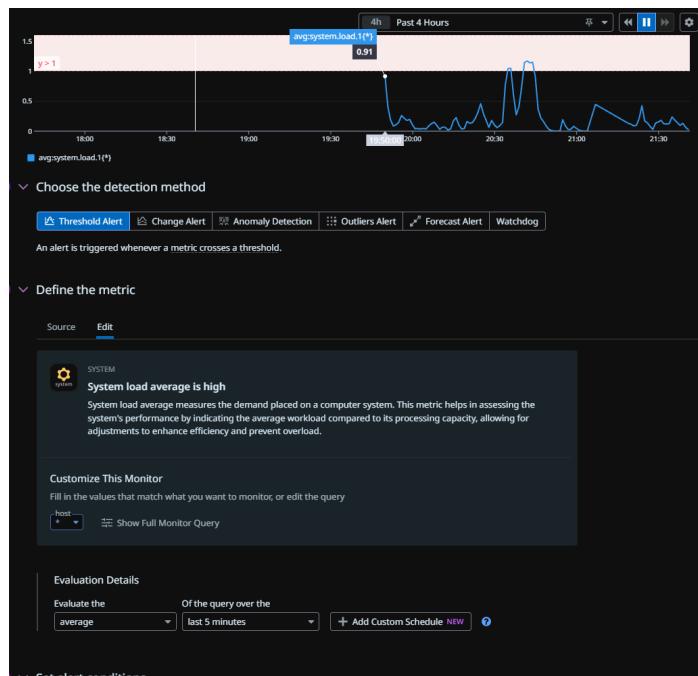
Cpu usage



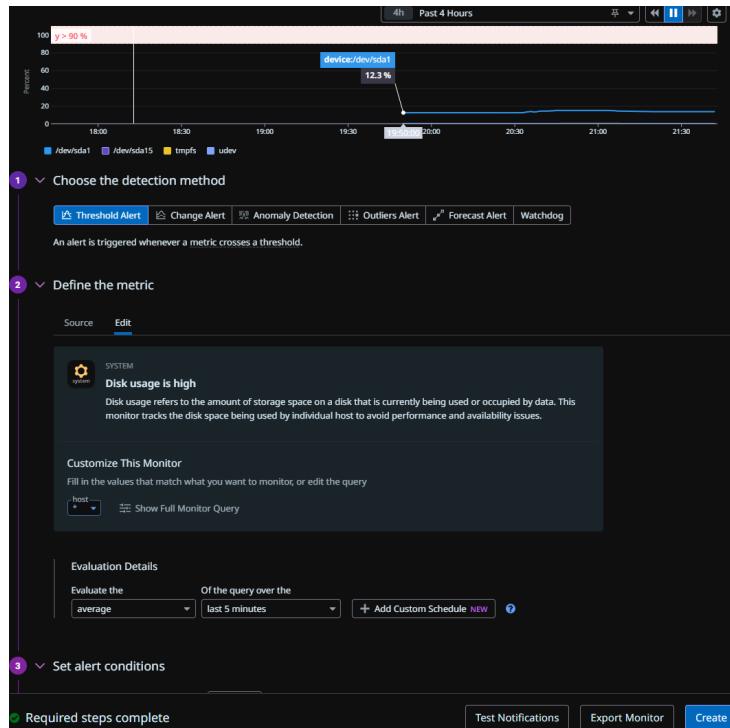
Memory space is low



System load average is high



Disk usage is high



Host OK monitor

1 Pick a Check

Select an integration : datadog.agent.up

2 Pick monitor scope

All Monitored Hosts excluding Select tags to exclude

3 Set alert conditions

Check Alert Cluster Alert An alert is triggered when consecutive run statuses cross your threshold.

Trigger a separate alert for each Select group reporting your check

Trigger the alert after selected consecutive failures:

Status: Warning

Status: Critical

Resolve the alert after selected consecutive successes:

Status: Ok

Notify if data is missing for more than 2 minutes.

(Never) automatically resolve this event from a no data state.

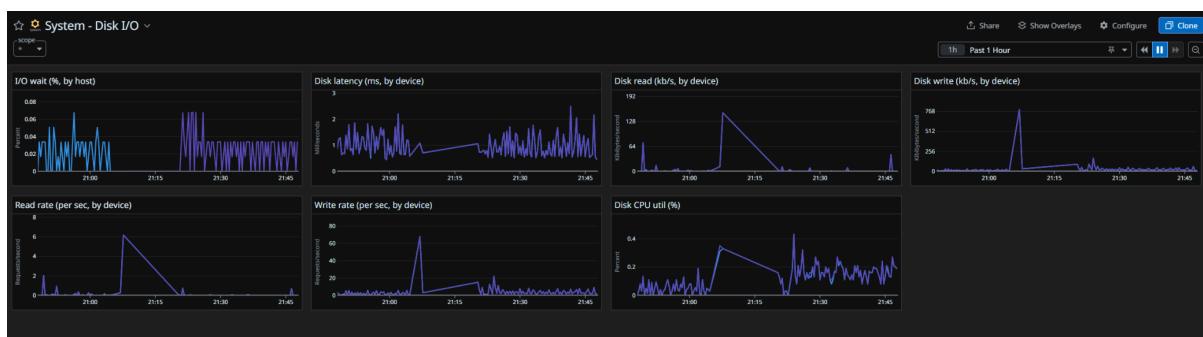
Required steps complete

Y si usas algunas de ellas y las guardas te saldrán de este modo pudiendo usarlas para poder recoger métricas ya que como he dicho anteriormente son **templates**.

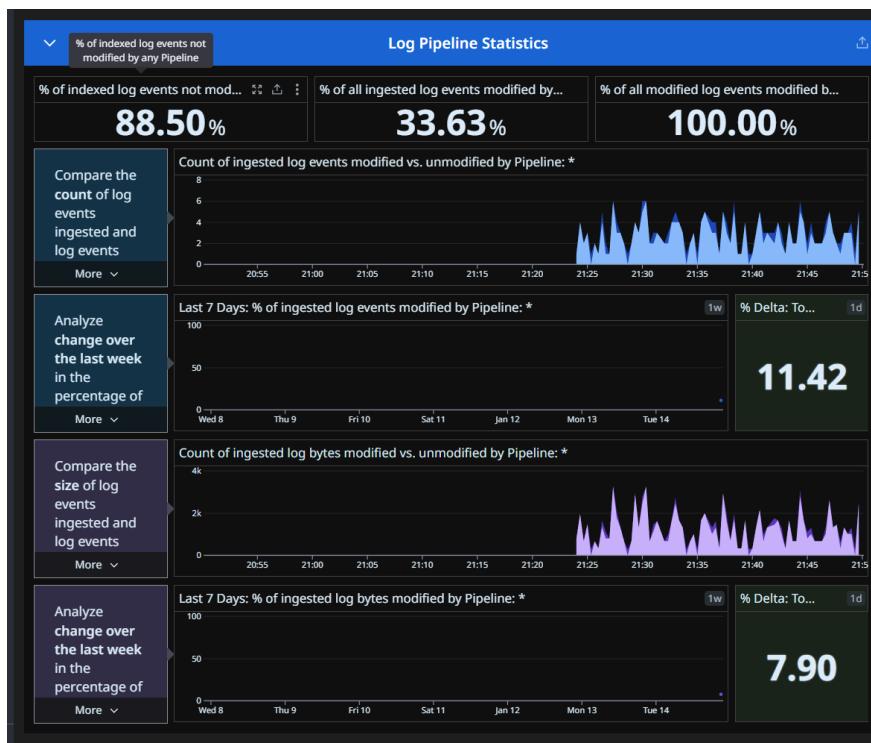
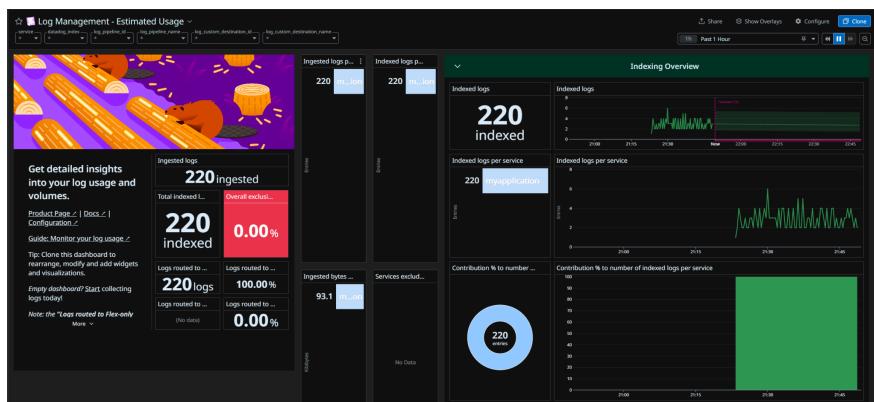
PRIORITY	STATUS	MUTED LEFT	NAME	TAGS
-	OK		System load is high for host *	
-	OK		Memory space is low for host *	
-	OK		Disk usage is high for host * on device {{device.name}}	
-	OK		CPU usage is high for host *	

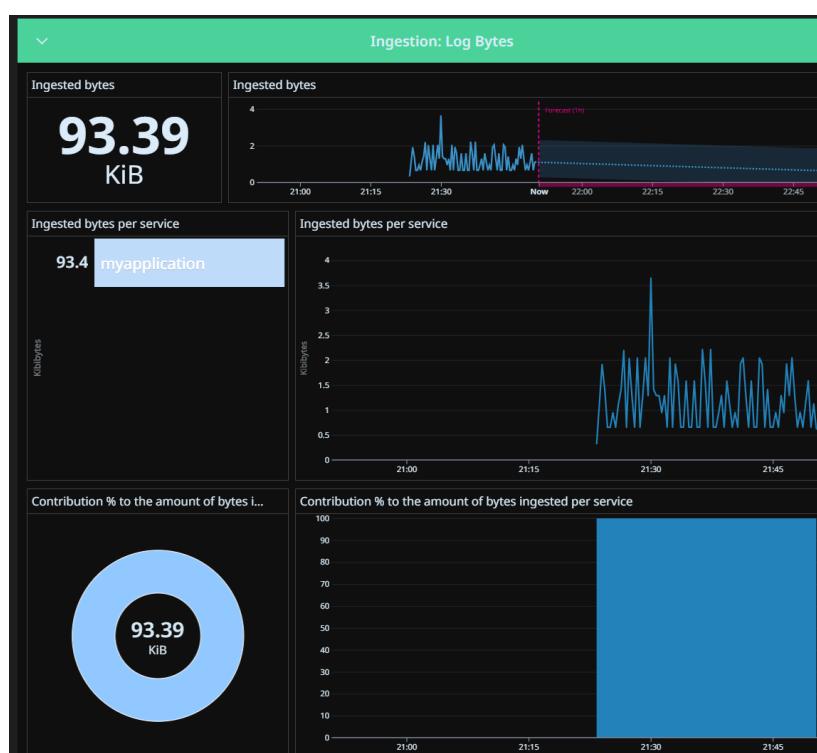
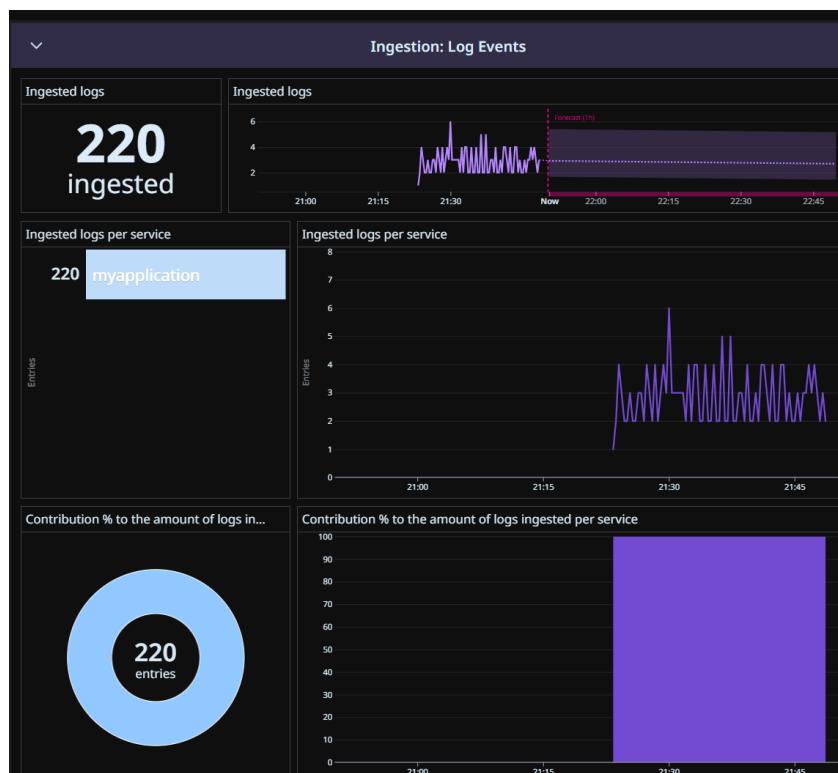
18º. Y por último también se pueden ver los **dashboards**

Si te metes en el **dashboard de system disk**



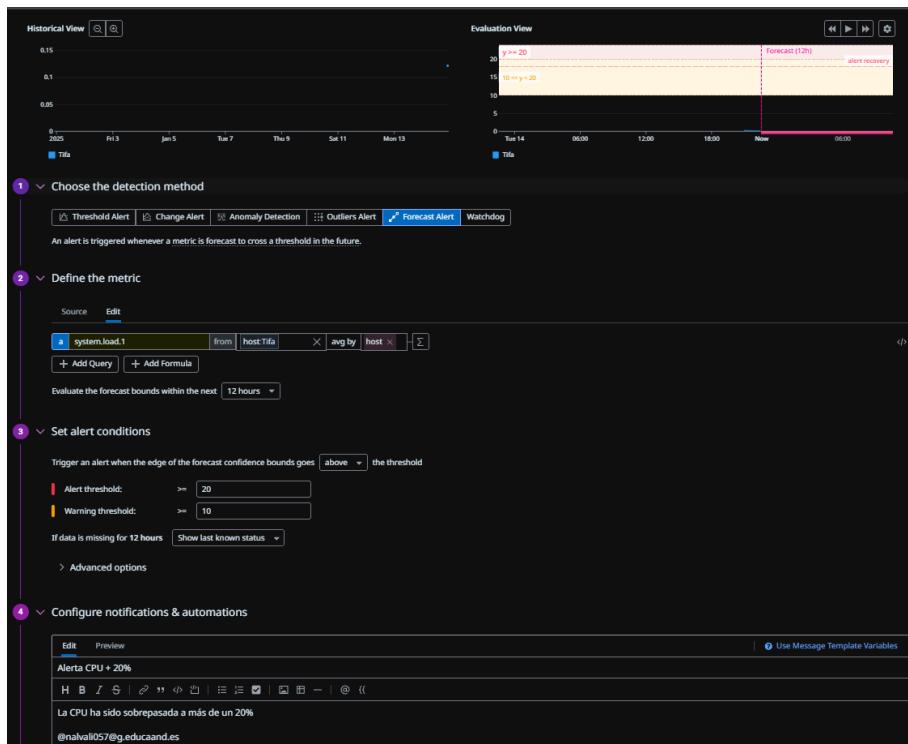
Y por último mostraré también los **dashboards** de logs (Que como se puede apreciar se recogen mediante un **pipeline**)



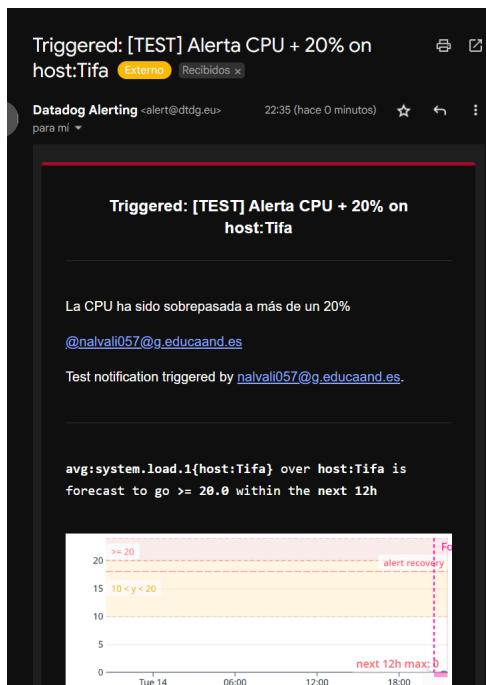


Alerta datadog

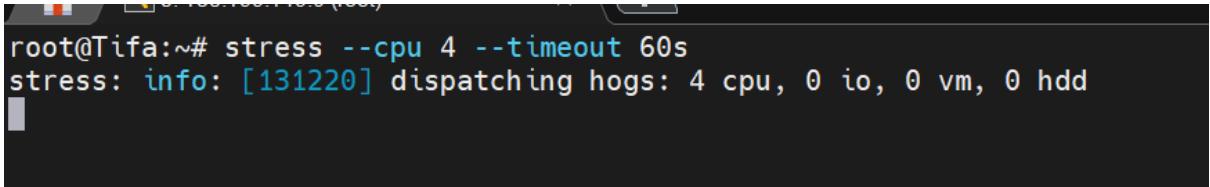
1º. Editaré una monitorización (en mi caso solo voy a crear la alerta de la CPU porque es lo más sencillo de implementar y nos sirve como ejemplo de uso)



2º. Probamos la alerta manualmente para ver si funciona (que hay que darle a **test**).

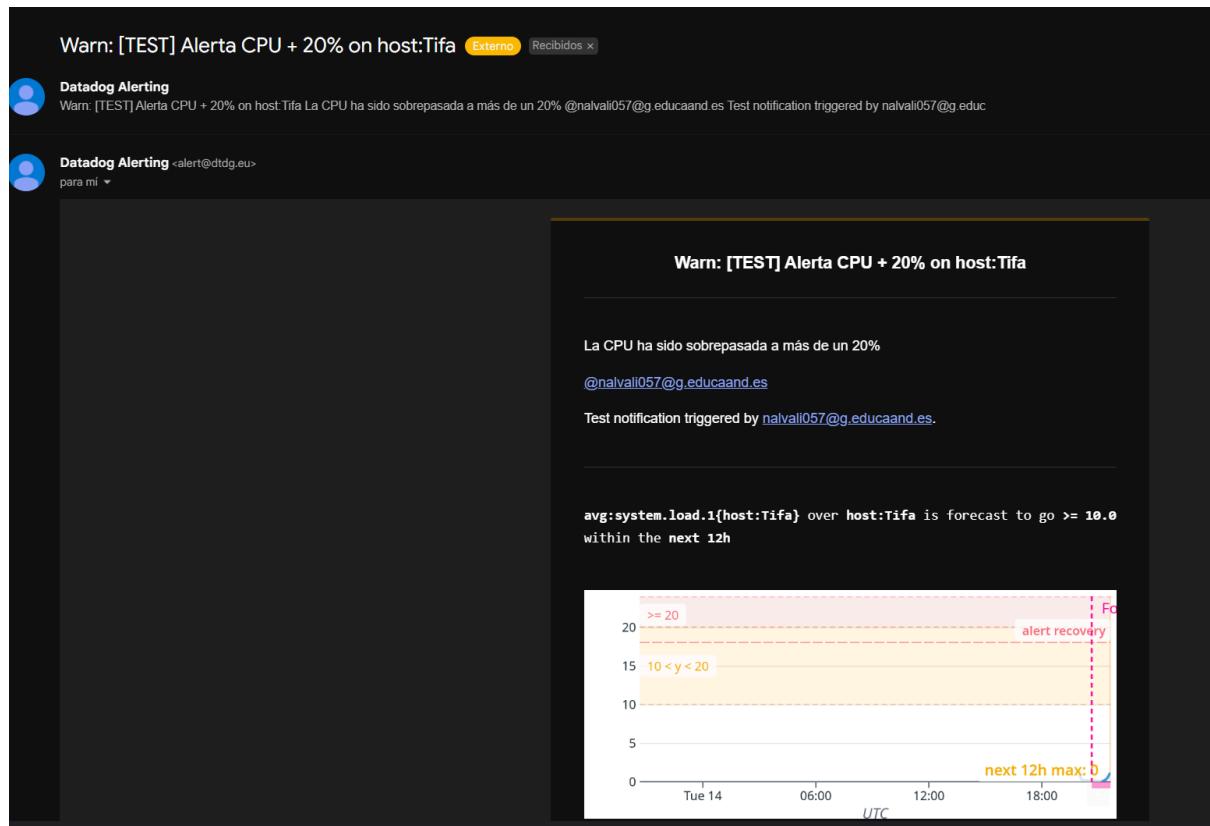


3º. Probamos a estresar la CPU de nuestro VPS.



```
root@Tifa:~# stress --cpu 4 --timeout 60s
stress: info: [131220] dispatching hogs: 4 cpu, 0 io, 0 vm, 0 hdd
```

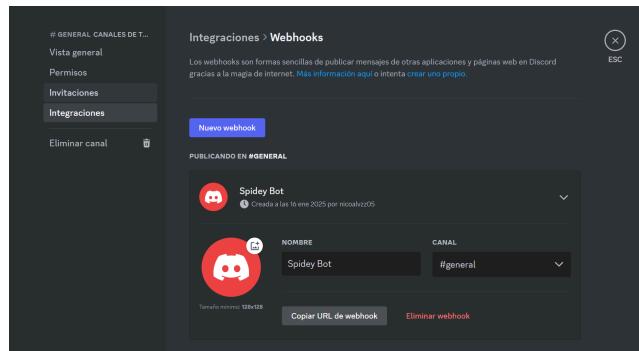
4º. Y por último veremos que me llega la alerta (Esto se debe a que la guardé anteriormente).



SCRIPTS AUTOMATIZACIÓN DE TAREAS

SCRIPT MONITOREO (ESTADO DE LOS NODOS DEL CLÚSTER CASSANDRA)

1º. Primero creamos un servidor para poder enviar las notificaciones del script.



2º. Creamos el script

```
#!/usr/bin/python
#Imports para solicitar requests
import requests

# configuración de acceso al VPS
SSH_HOST = "192.168.1.10"
SSH_PASSWORD = "Welcom"
CASSANDRA_HOST = "192.168.1.100.0"
DISCORD_WEBHOOK_URL = "https://discord.com/api/webhooks/132935961502932302/t23-q5K13kzj4H4fFPLsPh_cBVi-G0zr5_1Hsg9Rarv7Matta7FqoSCb_PV72K"

# Función para ejecutar un comando remoto usando SSH
def execute_remote_command(command):
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(CASSANDRA_HOST, username=SSH_USER, password=SSH_PASSWORD)
    stdin, stdout, stderr = ssh.exec_command(command)
    output = stdout.read().decode('utf-8')
    stderr = stderr.read().decode('utf-8')
    ssh.close()
    if stderr:
        raise Exception(f'Error: {stderr}')
    else:
        return output

# Función para obtener el estado de los nodos con nodetool
def get_nodetool_status():
    command = "nodetool status"
    remote_output = execute_remote_command(command)

# Función para analizar el estado de los nodos
def analyze_status(status_output):
    nodes = []
    lines = status_output.splitlines()
    for line in lines:
        if line.startswith("Datacenter") or line.startswith("status") or line.startswith("-"):
            continue
        # Dividir linea respetando espacios y asignar correctamente los campos
        parts = line.split()
        if len(parts) > 1:
            node = {}
            node['ip'] = parts[0] # Estado del nodo
            node['ip'] = parts[1] # Dirección IP
            node['rack'] = parts[2] # RACK
            node['tokens'] = parts[3] # Número de tokens
            node['partitions'] = parts[4] # Particiones
            node['host_id'] = parts[5] # ID de host
            node['rack'] = parts[6] # Rack
            nodes.append(node)
    return nodes
```

```
# Función para generar mensaje de notificación
def generate_message(nodes):
    if not nodes:
        return "No se pudieron obtener datos sobre los nodos de Cassandra."
    header = "Estado de los Nodos de Cassandra:\n"
    details = "\n"
    active = False
    for node in nodes:
        if node['status'] != 'UN':
            active = True
            details += f"- Estado: {node['status']}\n"
            details += f"- IP: {node['ip']}\n"
            details += f"- RACK: {node['rack']}\n"
            details += f"- Tokens: {node['tokens']}\n"
            details += f"- Particiones: {node['partitions']}\n"
            details += f"- Host ID: {node['host_id']}\n"
            details += f"- Rack: {node['rack']}\n"
    summary = "\nAlgunos nodos no están en estado 'UN'.\n"
    if active:
        summary += "Todos los nodos están activos y funcionando correctamente.\n"
    return header + summary + details

# Función para enviar notificación a Discord
def send_discord_notification(content):
    try:
        data = {"content": content}
        headers = {"Content-Type": "application/json"}
        response = requests.post(DISCORD_WEBHOOK_URL, json=data, headers=headers)
        print(f"Mensaje enviado a Discord con éxito.\n")
        if response.status_code != 200:
            print(f"Error al enviar notificación: {response.status_code}")
    except Exception as e:
        print(f"Error al enviar notificación: {str(e)}")

# Función principal para monitoreo de Cassandra
def monitor_cassandra():
    print("Monitoreando el estado de Cassandra...")
    status_output = get_cassandra_status()
    if status_output:
        print(f"Estado: {status_output}")
        discord_notification(status_output)
    else:
        print("No se pudieron obtener datos sobre los nodos de Cassandra.")

    nodes = analyze_status(status_output)
    message = generate_message(nodes)
    print(f"Mensaje enviado: {message}")
    print(f"Mensaje enviado: {message}")

    # Determinar si todos los nodos están en estado 'UN'
    if all(node['status'] == 'UN' for node in nodes):
        print(f"Todos los nodos están en estado 'UN'. Finalizando monitoreo.")
        return True
    return False

# Ejecutar monitoreo una sola vez
if __name__ == "__main__":
    if monitor_cassandra():
        print("Monitoreo finalizado. Verifica los problemas reportados.")
```

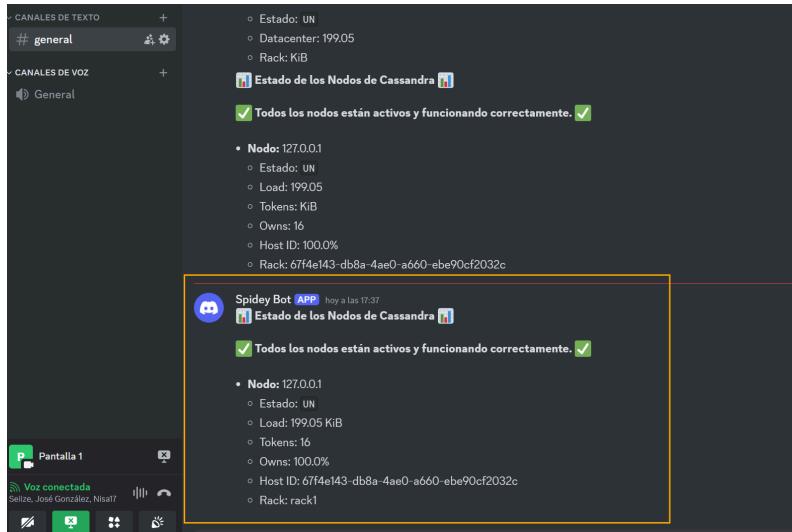
3º. Ejecutamos el script

```
root@Tifa:~# python3 MonitoreoAlertaDisc.py
🔍 Verificando estado de Cassandra...
☑ Notificación enviada a Discord con éxito.
📩 Mensaje enviado:
⬇ Estado de los Nodos de Cassandra** ⬇

☒ **Todos los nodos están activos y funcionando correctamente.** ☒
- **Nodo:** 127.0.0.1
- Estado: UN
- Load: 199.05 KiB
- Tokens: 16
- Owns: 100.0%
- Host ID: 67f4e143-db8a-4ae0-a660-ebe90cf2032c
- Rack: rack1

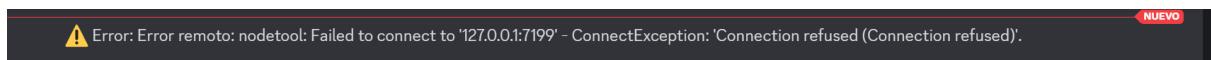
☒ Todos los nodos están en estado 'UN'. Finalizando monitoreo.
root@Tifa:~# nodetool status
Datacenter: datacenter1
=====
Status=Up/Down
/| State=Normal/Leaving/Joining/Moving
-- Address      Load      Tokens  Owns (effective) Host ID      Rack
UN 127.0.0.1    199.05   16        100.0%          67f4e143-db8a-4ae0-a660-ebe90cf2032c  rack1
root@Tifa:~#
```

4º. Verificamos la alerta



5º. Probaremos también con los nodos caídos.

```
root@Tifa:~# python3 MonitoreoAlertaDisc.py
🔍 Verificando estado de Cassandra...
⚠ Error: Error remoto: Failed to connect to '127.0.0.1:7199' - ConnectException: 'Connection refused (Connection refused)'.
☒ Notificación enviada a Discord con éxito.
⚠ Monitoreo no finalizado. Verifica los problemas reportados.
root@Tifa:~#
```



MIGRACIÓN BD

1º. Para migrar una bd tenemos que tener una Base de datos y una tabla con contenido.

```
cassandra@cqlsh:mi_keyspace> select * from usuarios;
 id | edad | email | nombre
---+---+---+---+
 9aad4c02-2714-4415-a0da-40d426525482 | 30 | juan.perez@example.com | Juan Pérez
 466bc95e-e8ee-46cd-85fd-91a64fdef24 | 40 | carlos.sanchez@example.com | Carlos Sánchez
 471f3589-7b76-47e6-ac20-55c2f402ccb4 | 25 | ana.garcia@example.com | Ana García
(3 rows)
cassandra@cqlsh:mi_keyspace>
```

2º. Creamos el script

```
#!/usr/bin/env python
# Función para conectar a Cassandra
def connect_to_cassandra(host, port, user, password):
    try:
        auth_provider = PlainTextAuthProvider(username=user, password=password)
        cluster = Cluster([host], port=port, auth_provider=auth_provider)
        session = cluster.connect()
        return session
    except Exception as e:
        print(f"Error al conectar a Cassandra: {e}")

# Función para obtener todos los registros de una tabla
def get_all_records(session, keyspace, table):
    try:
        query = f"SELECT * FROM ({keyspace}).({table})"
        result = session.execute(query)
        return result
    except Exception as e:
        print(f"Error al obtener datos de la tabla ({table}): {e}")
        return []

# Función para crear el keyspace y la tabla de respaldo
def create_backup_table(session):
    try:
        session.execute(f"CREATE KEYSPACE IF NOT EXISTS {DEST_KEYSPACE} WITH replication = ('class': 'SimpleStrategy', 'replication_factor': 1 )")
        session.execute(f"CREATE TABLE IF NOT EXISTS {DEST_KEYSPACE}.usuarios_backup (id UUID PRIMARY KEY, nombre TEXT, email TEXT, edad INT)")
        print(f"Keyspace {DEST_KEYSPACE} y tabla usuarios_backup creados correctamente")
    except Exception as e:
        print(f"Error al crear el keyspace o la tabla: {e}")

# Función para insertar registros en la base de datos de destino
def insert_records(session, keyspace, table, records):
    for record in records:
        columns = [record[i] for i in range(len(record) - 1)]
        values = [record[-1]] if isinstance(record[-1], str) else str(record[-1])
        query = f"INSERT INTO {keyspace}.({table}) ({columns}) VALUES ({values})"
        session.execute(query)
        print(f"Registros insertados correctamente en la tabla ({table})")
    except Exception as e:
        print(f"Error al insertar datos en la tabla ({table}): {e}")

# Función principal para migrar datos
def migrate_data():
    print("Iniciando la migración de datos de Cassandra a Cassandra...")

    # Conectar a la base de datos de origen
    source_session = connect_to_cassandra(SOURCE_CASSANDRA_HOST, SOURCE_CASSANDRA_PORT, CASSANDRA_USER, CASSANDRA_PASSWORD)
    if not source_session:
        return

    # Conectar a la base de datos de destino
    dest_session = connect_to_cassandra(DEST_CASSANDRA_HOST, DEST_CASSANDRA_PORT, CASSANDRA_USER, CASSANDRA_PASSWORD)
    if not dest_session:
        return

    # Crear el keyspace y la tabla de respaldo
    create_backup_table(dest_session)

    # Obtener las tablas disponibles en el keyspace
    try:
        source_session.set_keyspace(SOURCE_KEYSPACE)
        tables_query = f"SELECT table_name FROM system_schema.tables WHERE keyspace_name='{SOURCE_KEYSPACE}'"
        tables = source_session.execute(tables_query)
        for table in tables:
            table = table.table_name
            print(f"Migrando tabla: {table}")

        # Obtener todos los registros de la tabla en la base de datos de origen
        records = get_all_records(source_session, SOURCE_KEYSPACE, table)

        if records:
            # Insertar los registros en la base de datos de destino
            insert_records(dest_session, DEST_KEYSPACE, f"{table}_backup", records)

    except Exception as e:
        print(f"Error al obtener las tablas o migrar datos: {e}")

    print("Migración completada.")

if __name__ == "__main__":
    migrate_data()
```

```
# Función principal para migrar datos
def migrate_data():
    print("Iniciando la migración de datos de Cassandra a Cassandra...")

    # Conectar a la base de datos de origen
    source_session = connect_to_cassandra(SOURCE_CASSANDRA_HOST, SOURCE_CASSANDRA_PORT, CASSANDRA_USER, CASSANDRA_PASSWORD)
    if not source_session:
        return

    # Conectar a la base de datos de destino
    dest_session = connect_to_cassandra(DEST_CASSANDRA_HOST, DEST_CASSANDRA_PORT, CASSANDRA_USER, CASSANDRA_PASSWORD)
    if not dest_session:
        return

    # Crear el keyspace y la tabla de respaldo
    create_backup_table(dest_session)

    # Obtener las tablas disponibles en el keyspace
    try:
        source_session.set_keyspace(SOURCE_KEYSPACE)
        tables_query = f"SELECT table_name FROM system_schema.tables WHERE keyspace_name='{SOURCE_KEYSPACE}'"
        tables = source_session.execute(tables_query)
        for table in tables:
            table = table.table_name
            print(f"Migrando tabla: {table}")

        # Obtener todos los registros de la tabla en la base de datos de origen
        records = get_all_records(source_session, SOURCE_KEYSPACE, table)

        if records:
            # Insertar los registros en la base de datos de destino
            insert_records(dest_session, DEST_KEYSPACE, f"{table}_backup", records)

    except Exception as e:
        print(f"Error al obtener las tablas o migrar datos: {e}")

    print("Migración completada.")

if __name__ == "__main__":
    migrate_data()
```

3º. Lo ejecutamos

```
(mi_entorno) root@EzioCloud:~/mi_entorno/bin# python MigracionDB.py
Iniciando la migración de datos de Cassandra a Cassandra...
Keyspace backup y tabla usuarios_backup creados correctamente.
Migrando tabla: usuarios
Registros insertados correctamente en la tabla usuarios_backup
Migración completada.
(mi_entorno) root@EzioCloud:~/mi_entorno/bin#
```

```
cassandra@cqlsh:mi_keyspace> select * from usuarios;
+-----+-----+-----+
| id   | edad | email        | nombre |
+-----+-----+-----+
| 9aad4c02-2714-4415-a0da-40d426525482 | 30  | juan.perez@example.com | Juan Pérez
| 466bc95e-e8ee-46cd-85fd-91a64fdef24 | 40  | carlos.sanchez@example.com | Carlos Sánchez
| 471f3589-7b76-47e6-ac20-55c2f402ccb4 | 25  | ana.garcia@example.com | Ana García
(3 rows)
cassandra@cqlsh:mi_keyspace> use backup
... ;
cassandra@cqlsh:backup> select * from usuarios_backup;
+-----+-----+-----+
| id   | edad | email        | nombre |
+-----+-----+-----+
| 9aad4c02-2714-4415-a0da-40d426525482 | 30  | juan.perez@example.com | Juan Pérez
| 466bc95e-e8ee-46cd-85fd-91a64fdef24 | 40  | carlos.sanchez@example.com | Carlos Sánchez
| 471f3589-7b76-47e6-ac20-55c2f402ccb4 | 25  | ana.garcia@example.com | Ana García
(3 rows)
cassandra@cqlsh:backup> exit
root@Tifa:~# cqlsh 138.199.149.0 -u cassandra -p cassandra
Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.
Connected to Test Cluster at 138.199.149.0:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> describe keyspaces;
+-----+
| backup      system_auth      system_traces
| mi_keyspace system_distributed system_views
| system      system_schema     system_virtual_schema
+-----+
cassandra@cqlsh> use backup;
cassandra@cqlsh:backup> describe tables;
+-----+
| usuarios_backup
+-----+
cassandra@cqlsh:backup>
```

COPIA DE SEGURIDAD BD (EN BASH)

1º. Creamos el script

```
GNU nano 8.3                                         CopiaSeguridadDB.sh

#!/bin/bash

# Configuración de Cassandra
CASSANDRA_USER="cassandra"
CASSANDRA_PASSWORD="cassandra"
CASSANDRA_HOST="138.199.149.0"
KEYSPACE="mi_keyspace"
BACKUP_DIR="/BackupBD"
DATE=$(date +%Y%m%d%H%M)

# Crear un archivo de backup
nodetool -h $CASSANDRA_HOST -u $CASSANDRA_USER -pw $CASSANDRA_PASSWORD snapshot $KEYSPACE -t $DATE

# Mover el snapshot a la carpeta de backup
if [ -d "/var/lib/cassandra/data/$KEYSPACE" ]; then
    mv /var/lib/cassandra/data/$KEYSPACE/*/*snapshots/$DATE $BACKUP_DIR/$KEYSPACE-$DATE
else
    echo "El keyspace $KEYSPACE no existe o no tiene datos."
fi

# Eliminar snapshots antiguos (más de 7 días)
find $BACKUP_DIR -type d -name "$KEYSPACE-*" -mtime +7 -exec rm -r {} \;
```

2º. Ejecutamos el script

```
root@Tifa:~# cqlsh 138.199.149.0 -u cassandra -p cassandra
root@Tifa:~/# ./CopiaSeguridadDB.sh

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Using 1 child processes

Starting copy of mi_keyspace.usuarios with columns [id, edad, email, nombre].
Processed: 3 rows; Rate:      90 rows/s; Avg. rate:      8 rows/s
3 rows exported to 1 files in 0.390 seconds.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

<stdin>:1:Improper COPY command.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

<stdin>:1:Improper COPY command.
Copia de seguridad completada.
```

```
root@Tifa:~/# ls /BackupBD/
mi_keyspace-202501161749.cql  mi_keyspace-usuarios-202501161749.csv
mi_keyspace-202501161750.cql
root@Tifa:~/#
```

RESTAURACIÓN COPIA BD (EN BASH)

1º. Creamos un script para restaurar una copia concreta

```
GNU nano 8.3
#!/bin/bash

# Configuración de Cassandra
CASSANDRA_USER="cassandra"
CASSANDRA_PASSWORD="cassandra"
CASSANDRA_HOST="138.199.149.0"
KEYSPACE="mi_keyspace"
BACKUP_DIR="/BackupBD"
DATE="`date +\%Y\%m\%d\%H\%M\%S`"
BACKUP_FILE="$BACKUP_DIR/$KEYSPACE-$DATE.cql"

# Restaurar el esquema
cqlsh $CASSANDRA_HOST -u $CASSANDRA_USER -p $CASSANDRA_PASSWORD -f $BACKUP_FILE

# Importar los datos de las tablas
TABLES=$cqlsh $CASSANDRA_HOST -u $CASSANDRA_USER -p $CASSANDRA_PASSWORD -e "SELECT table_name FROM system_schema.tables WHERE keyspace_name='$KEYSPACE';" | awk '{NR>3 {print $1}}'

for table in $TABLES; do
    if [ "$table" != "table_name" ] && [ "$table" != "" ]; then
        cqlsh $CASSANDRA_HOST -u $CASSANDRA_USER -p $CASSANDRA_PASSWORD -e "COPY $KEYSPACE.$table FROM '$BACKUP_DIR/$KEYSPACE-$table-$DATE.csv' WITH HEADER = TRUE;"
    fi
done

echo "Restauración completada."
```

2º. Eliminamos la BD y ejecutamos el script, se verá que se crea de nuevo la bd.

```
root@Tifa:~# cqlsh 138.199.149.0 -u cassandra -p cassandra
Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Connected to Test Cluster at 138.199.149.0:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> DROP KEYSPACE mi_keyspace;
cassandra@cqlsh> exit
root@tifa:~/
```

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Using 1 child processes

Starting copy of mi_keyspace.usuarios with columns [id, edad, email, nombre].
Processed: 3 rows; Rate: 5 rows/s; Avg. rate: 7 rows/s
3 rows imported from 1 files in 0.420 seconds (0 skipped).

Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

<stdin>:1:Improper COPY command.
Restauración completada.

```
root@Tifa:~# cqlsh 138.199.149.0 -u cassandra -p cassandra
Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

Connected to Test Cluster at 138.199.149.0:9042
[cqlsh 6.1.0 | Cassandra 4.1.7 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> use mi_keyspace;
cassandra@cqlsh> select * from usuarios;
```

id	edad	email	nombre
9aad4c02-2714-4415-a0da-40d426525482	30	juan.perez@example.com	Juan Pérez
466bc95e-e8ee-46cd-85fd-91a64fdefd24	40	carlos.sanchez@example.com	Carlos Sánchez
471f3589-7b76-47e6-ac20-55c2f402ccb4	25	ana.garcia@example.com	Ana García

(3 rows)

cassandra@cqlsh:mi_keyspace> ■

INVENTARIO DE EQUIPO

1º. Crearé un script para saber todas las tablas y sus descripciones.

```
GNU nano 7.2
InventariosEquipos.py

from cassandra.cluster import Cluster
from cassandra.auth import PlainTextAuthProvider

# Configuración de acceso a Cassandra
CASSANDRA_USER = "cassandra"
CASSANDRA_PASSWORD = "cassandra"
CASSANDRA_HOST = "138.199.149.0"
KEYSPACE = "m_keystore"

# Conectar a Cassandra
auth_provider = PlainTextAuthProvider(username=CASSANDRA_USER, password=CASSANDRA_PASSWORD)
cluster = Cluster([CASSANDRA_HOST], auth_provider=auth_provider)
session = cluster.connect()

# Obtener la lista de tablas
query_tables = f"SELECT table_name FROM system_schema.tables WHERE keyspace_name='{KEYSPACE}'"
tables = session.execute(query_tables)

# Guardar resultados en un archivo
with open('inventario.txt', 'w') as f:
    for table in tables:
        table_name = table.table_name
        f.write(f'Tabla: {table_name}\n')

# Obtener detalles de las columnas
query_columns = f"SELECT column_name, type FROM system_schema.columns WHERE keyspace_name='{KEYSPACE}' AND table_name='{table_name}'"
columns = session.execute(query_columns)
f.write(" Columnas:\n")
for column in columns:
    f.write(f" - {column.column_name} ({column.type})\n")

# Contar el número de filas
query_count = f"SELECT COUNT(*) FROM {KEYSPACE}.{table_name}"
count = session.execute(query_count).one()[0]
f.write(f" Número de filas: {count}\n\n")

print("Inventario completado.")


```

2º. Ejecutaré el script y veré el contenido del fichero.

```
(mi_entorno) root@EzioCloud:~/mi_entorno/bin# python3 InventariosEquipos.py
Inventario completado.
(mi_entorno) root@EzioCloud:~/mi_entorno/bin# ls
activate      Activate.ps1  InventariosEquipos.py  pip      python
activate.csh   cqlsh       inventario.txt        pip3    python3
activate.fish   geomet     MigracionDB.py      pip3.11  python3.11
(mi_entorno) root@EzioCloud:~/mi_entorno/bin# cat inventario.txt
Tabla: clientes
Columnas:
- dirección (text)
- email (text)
- id (uuid)
- nombre (text)
- teléfono (text)
Número de filas: 0

Tabla: detalles_pedidos
Columnas:
- cantidad (int)
- pedido_id (uuid)
- precio_unitario (decimal)
- producto_id (uuid)
Número de filas: 0

Tabla: pedidos
Columnas:
- cliente_id (uuid)
- fecha (timestamp)
- id (uuid)
- total (decimal)
Número de filas: 0

Tabla: productos
Columnas:
- descripción (text)
- id (uuid)
- nombre (text)
- precio (decimal)
- stock (int)
Número de filas: 0

Tabla: proveedores
Columnas:
- contacto (text)
- dirección (text)
- id (uuid)
- nombre (text)
- teléfono (text)
Número de filas: 0

Tabla: usuarios
Columnas:
- edad (int)
- email (text)
- id (uuid)
- nombre (text)
Número de filas: 3

(mi_entorno) root@EzioCloud:~/mi_entorno/bin#
```

ALERTA DISCORD MONITORIZACIÓN SISTEMA (NODOS DE CASSANDRA)

1º. Creamos el script

```
GNU nano 7.2
from cassandra.auth import PlainTextAuthProvider
from cassandra.cluster import Cluster
import psutil
import datetime
import requests

def generar_reporte():
    # Credenciales de Cassandra
    auth_provider = PlainTextAuthProvider(username='cassandra', password='cassandra')

    # Conectar a Cassandra
    cluster = Cluster(['138.199.149.0'], auth_provider=auth_provider)
    session = cluster.connect('system')

    # Obtener el estado de los nodos
    rows = session.execute("SELECT * FROM local")
    estados_nodos = ""
    for row in rows:
        # Cambiado el acceso a los valores por indices
        estados_nodos += f'Nodo {row[0]} - Estado: {row[1]}\n'

    # Obtener información del sistema
    cpu_percent = psutil.cpu_percent()
    memory_percent = psutil.virtual_memory().percent
    disk_usage = psutil.disk_usage('/').percent

    # Crear el reporte
    reporte = f"""
Reporte de Cassandra - {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}
-----
Uso de CPU: {cpu_percent}%
Uso de Memoria: {memory_percent}%
Uso de Disco: {disk_usage}%

Estado de los Nodos en Cassandra:
{estados_nodos}
"""

    # Enviar el reporte a Discord
    webhook_url = "https://discord.com/api/webhooks/1329355901592932362/t23-q5KI3kzjuHHUFPRLLsFh_oC8Vr-G0zs5_1Hamgj9Umrt47Uuttm7F0oSChb_PV7zK"
    data = {
        "content": reporte
    }
    try:
        response = requests.post(webhook_url, json=data)
        if response.status_code == 200 or response.status_code == 204:
            print("✅ Notificación enviada a Discord con éxito.")
        else:
            print(f"⚠️ Error al enviar notificación: {response.status_code}")
    except Exception as e:
        print(f"❌ Error al intentar enviar la notificación: {e}")

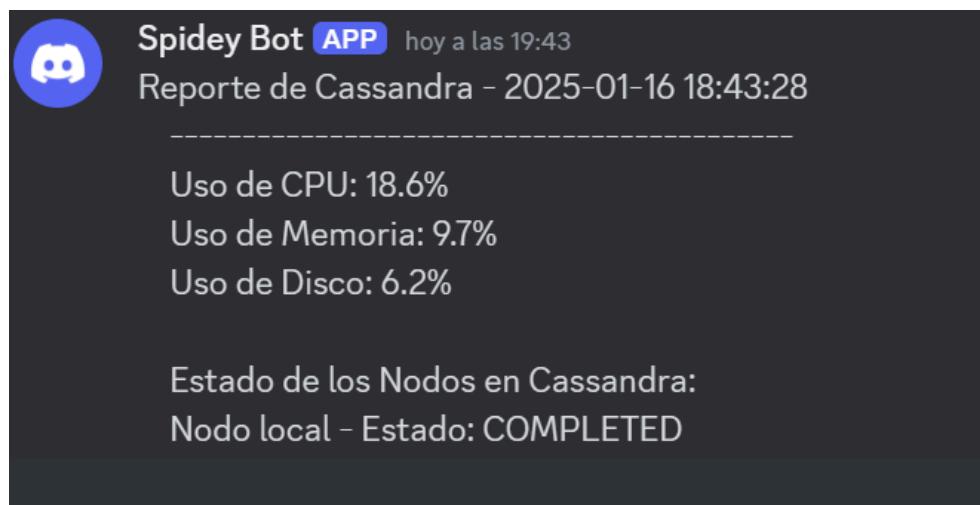
    # Guardar el reporte en un archivo
    with open(f"reporte_{datetime.datetime.now().strftime('%Y%m%d')}.txt", 'w') as file:
        file.write(reporte)

    print("Reporte generado con éxito.")

if __name__ == "__main__":
    generar_reporte()
```

2º. Ejecutamos el script

```
(mi_entorno) root@EzioCloud:~/mi_entorno/bin# python DeteccionProblemas.py
✅ Notificación enviada a Discord con éxito.
Reporte generado con éxito.
```



OPTIMIZACIÓN DEL RENDIMIENTO

INFORMACIÓN IMPORTANTE ACERCA DE ÍNDICES CASSANDRA

Las consultas en Cassandra son débiles pero son compatibles con la indexación y clasificación. Esto se debe al alto rendimiento y costo por lo que no se recomienda muchas veces el uso de estos a no ser que sea totalmente necesario.

Admite la creación de índices secundarios que pueden crearse en todas las columnas menos en las claves primarias.

```
CREATE TABLE test(
    a INT,
    b INT,
    c INT,
    d INT,
    e INT,
    m INT,
    PRIMARY KEY(a,b,c));

CREATE INDEX ON test(c);
CREATE INDEX ON test(e);
```

```
SELECCIONAR * DE la prueba DONDE e = 1; // Sí
SELECCIONE * DE la prueba DONDE e> 1; // No funcionará.
```

PORQUE NO SE RECOMIENDA EL USO DE ÍNDICES SECUNDARIOS

No se aconseja usar índices secundarios en Cassandra porque al ser una **base de datos distribuida**, las consultas que utilizan estos índices requieren escanear múltiples nodos para encontrar los datos, lo que introduce una alta latencia, especialmente en tablas grandes o distribuidas. Además, los índices generan una carga adicional en las escrituras, ya que deben actualizarse cada vez que se modifica la tabla, lo que degrada el rendimiento general. Estas limitaciones hacen que los índices secundarios no sean eficientes ni escalables en la mayoría de los casos, siendo preferible rediseñar el modelo de datos o usar tablas específicas para optimizar las consultas.

CASO PRÁCTICO (CLÚSTER EN GOOGLE CLOUD PLATFORM Y MONITORIZACIÓN DE RECURSOS CON PROMETHEUS, FLUENTBIT Y GRAFANA)

INTRODUCCIÓN

Antes de hacer el caso práctico explicaré que quiero hacer, quiero montar un clúster de **Cassandra** usando **google cloud platform**, una vez creado el clúster **quiero enseñar todas las funcionalidades que trae por defecto además de la monitorización de recursos.**

¿QUÉ ES Y PARA QUÉ SIRVE GOOGLE CLOUD Y LAS HERRAMIENTAS POR DEFECTO MENCIONADAS ANTERIORMENTE ?

Google Cloud es una plataforma de computación en la nube que proporciona una amplia gama de servicios y herramientas para **construir, desplegar y escalar aplicaciones y soluciones empresariales.** En lugar de depender de infraestructura local, puedes utilizar Google Cloud para acceder a recursos como **máquinas virtuales, almacenamiento, bases de datos, análisis de datos y servicios de inteligencia artificial,** pagando solo por lo que usas.

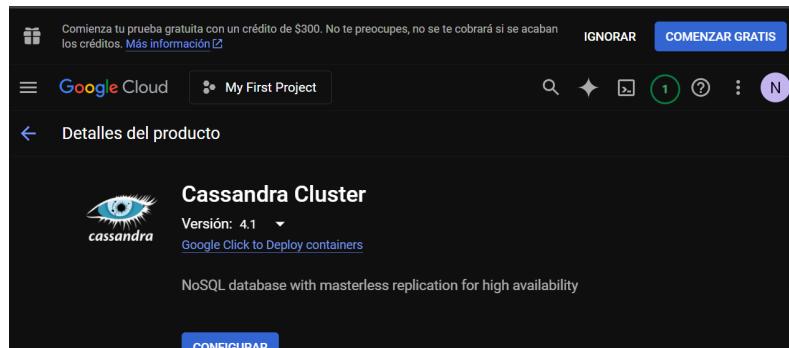
Google Cloud es ideal para desarrollar aplicaciones modernas, alojar sitios web, almacenar y analizar grandes volúmenes de datos, y ejecutar cargas de trabajo avanzadas, como aprendizaje automático. Entre sus productos destacados se incluyen **Compute Engine** (máquinas virtuales), **Cloud Storage** (almacenamiento escalable) y **BigQuery** (análisis de datos a gran escala). Estas herramientas son ampliamente utilizadas en sectores como tecnología, retail y finanzas, gracias a su flexibilidad, escalabilidad y capacidad de integrar inteligencia artificial en los procesos de negocio.

Prometheus, Fluent Bit y Grafana son herramientas fundamentales para la monitorización y observabilidad, especialmente en entornos distribuidos como un clúster de Cassandra, donde se instalan por defecto para gestionar el monitoreo. **Prometheus** es un sistema de monitoreo y alerta que recopila métricas de los nodos y las bases de datos, permitiendo analizar el rendimiento y detectar problemas. **Fluent Bit** es un colector de logs ligero que procesa y envía los registros generados por Cassandra y sus componentes, facilitando la trazabilidad y depuración. **Grafana** es una herramienta de visualización que utiliza los datos de Prometheus para generar paneles interactivos, ofreciendo una vista clara y en tiempo

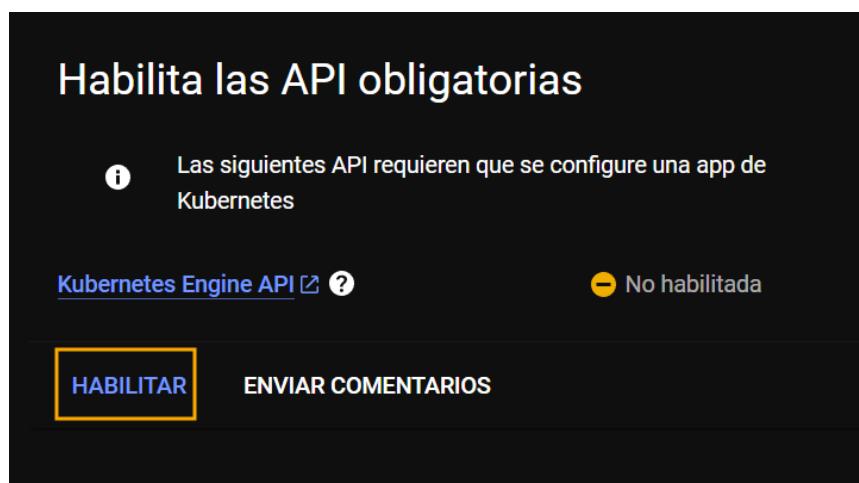
real del estado del clúster. Juntas, estas herramientas aseguran un monitoreo eficaz, proporcionando métricas, logs y visualizaciones que permiten mantener la estabilidad y el rendimiento del sistema.

PASOS A SEGUIR PARA DESPLEGAR UN CLÚSTER DE CASSANDRA EN GOOGLE CLOUD

1º. Accedemos a google cloud platform y buscamos en la barra superior **cassandra**.



2º. Al darle a configurar, nos saldrá una **API obligatoria**, esta trata de una api que requiere la configuración de una app de Kubernetes, es decir lo que vamos a montar que es el Clúster de Cassandra va a ser montado por Kubernetes.



3º. Introducimos la localización, red y réplicas que queramos tener en nuestro clúster.

HAZ CLIC PARA IMPLEMENTAR EN GKE > **IMPLEMENTAR MEDIAI**

Información adicional

Tu app usará instancias de procesamiento administradas en una agrupación lógica llamada "clúster", cuya configuración es perfecta para comenzar a usar Kubernetes. Si quieras conocer más opciones, visita la [página de creación de clústeres de Kubernetes Engine](#).

Zona: us-central1-a

Red: default

Subred: default

CREAR UN NUEVO CLÚSTER O SELECCIONAR UN CLÚSTER EXISTENTE

Espacio de nombres: default

El espacio de nombres en el que se implementará la aplicación

Nombre de la instancia de app *: cassandra-p20

Replicas *: 3

StorageClass: Create a new storage class

Storage size for persistent volumes: 5Gi

Enable Stackdriver Metrics Exporter

IMPLEMENTAR

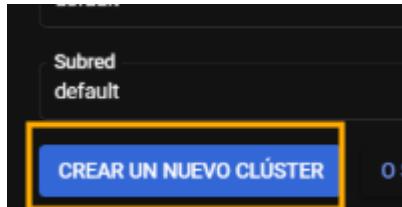
Descripción general de Cassandra Cluster
Solución proporcionada por Google Click to Deploy containers

Precios
Nota: No se cobra ninguna tarifa por uso de este producto. Se aplicarán cargos por el uso de Google Kubernetes Engine. Consulta la [lista de precios de Google Cloud](#) para conocer los precios más recientes.
Todos los productos tienen precios en USD y se cobran en la moneda () especificada en tu cuenta de facturación. El precio de este mes se calcula con un tipo de cambio de USD 1 = 0

Documentación

- [User Guide](#) Get started with Google Cloud Platform's Cassandra Kubernetes application
- [DataStax Cassandra documentation](#) Official documentation for Cassandra 3.0 provided by DataStax.
- [Apache documentation for Cassandra](#) Documentation hosted by Apache Software Foundation.

4º. Le damos a crear un nuevo clúster cuando tengamos la configuración ya completada.



5º. Y posteriormente a **implementar**.

HAZ CLIC PARA IMPLEMENTAR EN GKE > **IMPLEMENTAR MEDIAI**

Información adicional

Tu app usará instancias de procesamiento administradas en una agrupación lógica llamada "clúster", cuya configuración es perfecta para comenzar a usar Kubernetes. Si quieras conocer más opciones, visita la [página de creación de clústeres de Kubernetes Engine](#).

Se creó correctamente el clúster "cluster-1" en la zona "us-central1-a".

Espacio de nombres: default

El espacio de nombres en el que se implementará la aplicación

Nombre de la instancia de app *: cassandra-p20

Replicas *: 3

StorageClass: premium-rwo

Storage size for persistent volumes: 5Gi

Enable Stackdriver Metrics Exporter

IMPLEMENTAR

Descripción general de Cassandra Cluster
Solución proporcionada por Google Click to Deploy containers

Precios
Nota: No se cobra ninguna tarifa por uso de este producto. Se aplicarán cargos por el uso de Google Kubernetes Engine. Consulta la [lista de precios de Google Cloud](#) para conocer los precios más recientes.
Todos los productos tienen precios en USD y se cobran en la moneda () especificada en tu cuenta de facturación. El precio de este mes se calcula con un tipo de cambio de USD 1 = 0

Documentación

- [User Guide](#) Get started with Google Cloud Platform's Cassandra Kubernetes application
- [DataStax Cassandra documentation](#) Official documentation for Cassandra 3.0 provided by DataStax.
- [Apache documentation for Cassandra](#) Documentation hosted by Apache Software Foundation.

6º. Esperamos que se implemente el clúster.

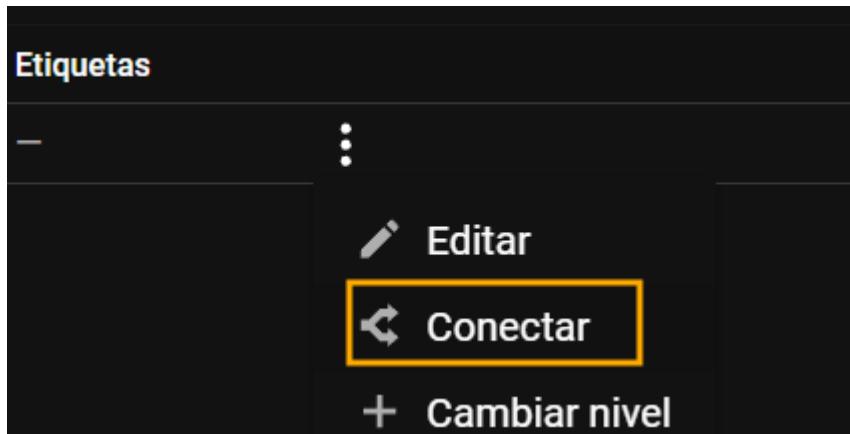
The screenshot shows the 'Kubernetes Engine' interface. On the left, a sidebar lists resources: 'Más información sobre Ente...', 'Todas las flotas', 'Administración de recursos' (with 'Descripción general', 'Clústeres', 'Cargas de trabajo', 'Equipos', and 'Aplicaciones' selected), 'IA/AA NUEVO', 'Secrets y ConfigMaps', and 'Almacenamiento'. The main area displays the deployment of a 'cassandra-p20' application, with a progress bar indicating 'Implementando componentes de la aplicación...' and a status message 'Implementando aplicación... listo'. A link 'Vista previa de la instantánea de la aplicación' is visible at the bottom.

7º. Una vez se haya implementado comprobamos que el clúster está activo

The screenshot shows the 'Google Cloud' interface with 'My First Project'. In the top navigation, there's a search bar for 'cassandra cluster'. The main view is 'Detalles de la aplicación' for the 'cassandra-p20' cluster. It shows the cluster status as 'OK' with a green checkmark. The 'DETAILS' tab is selected, displaying information like 'Cluster: cluster-1', 'Espacio de nombres: default', 'Fecha de creación: 19 ene 2025 14:25:23', and 'Etiquetas: app.kubernetes.io/name: cassandra-p20'. The 'COMPONENTES' tab lists components: a Stateful Set named 'cassandra-p20-cassandra' with 4 OK pods, three Persistent Volume Claims (cassandra-p20-cassandra-pvc-cassandra-p20-cassandra-0, cassandra-p20-cassandra-pvc-cassandra-p20-cassandra-1, cassandra-p20-cassandra-pvc-cassandra-p20-cassandra-2) all in 'Bound' state, and a Service named 'cassandra-p20-cassandra-svc' with 4 OK pods.

The screenshot shows a configuration wizard for GKE. It starts with 'Ocho pasos para configurar GKE' and an 'INICIAR' button. Below this, it says 'Ejecuta tus cargas de trabajo esenciales de forma más rápida, segura y fácil a escala empresarial'. It explains that GKE Enterprise combines multiple clusters and workloads with security, management, and network tools. A section titled 'APRENDER Y HABILITAR' follows. At the bottom, there's a table with tabs for 'DESCRIPCIÓN GENERAL', 'OBSERVABILIDAD', and 'OPTIMIZACIÓN DE COSTOS'. The 'DESCRIPCIÓN GENERAL' tab shows a single cluster entry: 'cluster-1' (Estado: OK, Nombre: cluster-1, Ubicación: us-central1-a, Nivel: Estándar, Cantidad de nodos: 4, CPU virtuales totales: 8, Memoria total: 27 GB). There's also a 'Filtro' input field.

8º. También podemos conectarnos a la consola por ssh.



Para ello después de darle a conectar le podemos dar a **Ejecutar En Cloud Shell**.

Conéctate al clúster

Puedes realizar la conexión al clúster mediante la línea de comandos o con un panel.

Acceso a la línea de comandos

Configura el acceso a la línea de comandos de [kubectl](#) ejecutando el siguiente comando:

```
$ gcloud container clusters get-credentials cluster-1 --zone us-central1-a -- project airy-office-448312-g8
```

EJECUTAR EN CLOUD SHELL

Panel de Cloud Console

Puedes ver las cargas de trabajo que se encuentran en ejecución en tu clúster en el [panel de cargas de trabajo](#) de Cloud Console.

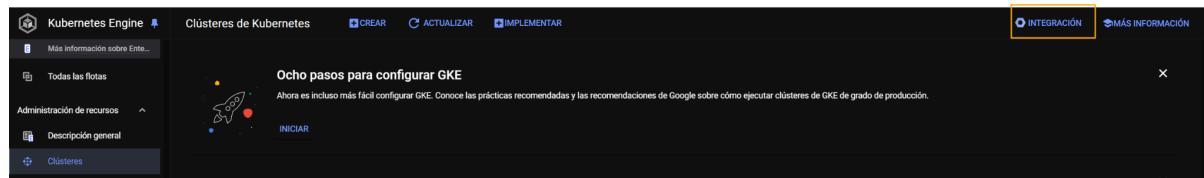
[ABRIR EL PANEL DE CARGAS DE TRABAJO](#)

ACEPTAR

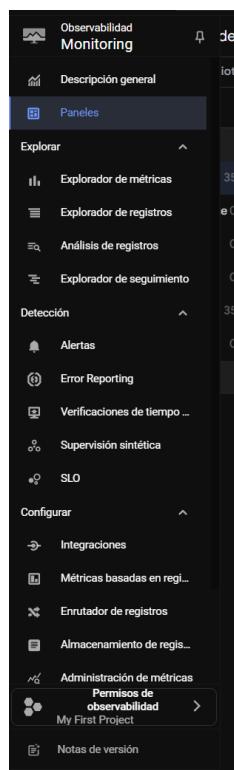
9º. Accedemos a la máquina que contiene **Cassandra** (En esta puedes acceder de forma local con **CQLSH** aunque en mi caso no he hecho captura de esto)

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to airy-office-448312-g8.
Use `gcloud config set project [PROJECT_ID]` to change to a different project.
cloudshell:~ (airy-office-448312-g8)$ gcloud container clusters get-credentials cluster-1 --zone us-central1-a --project airy-office-448312-g8nicoalvaali@cloudshell:~ (airy-office-448312-g8)$
Fetching cluster endpoint and auth data.
kubeconfig entry generated for cluster-1.
nicoalvaali@cloudshell:~ (airy-office-448312-g8)$
```

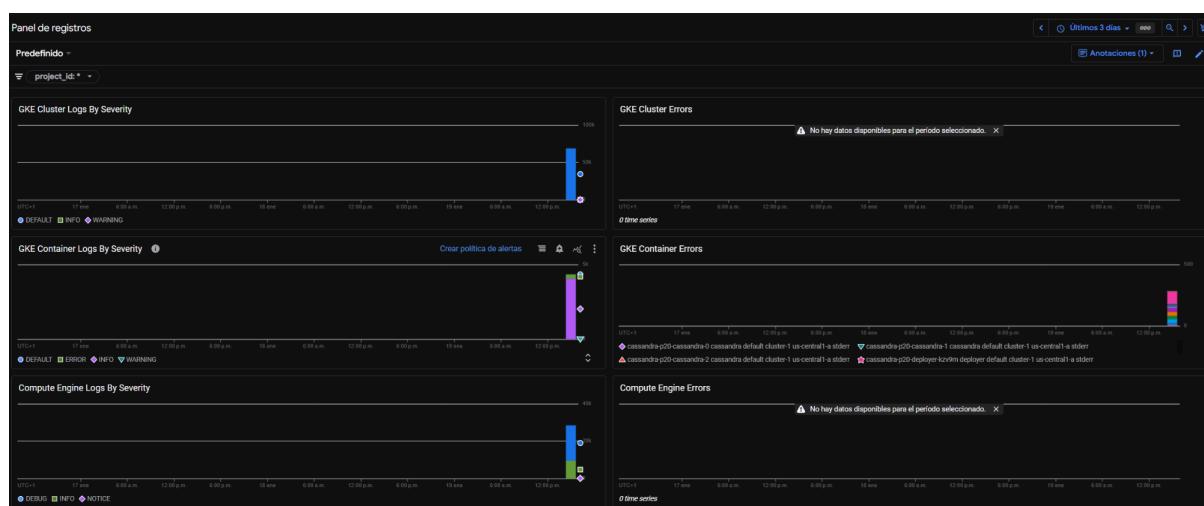
10º. Si queremos monitorizar vamos a configuración y le damos a monitorización



Que como se puede apreciar hay todas las opciones siguientes



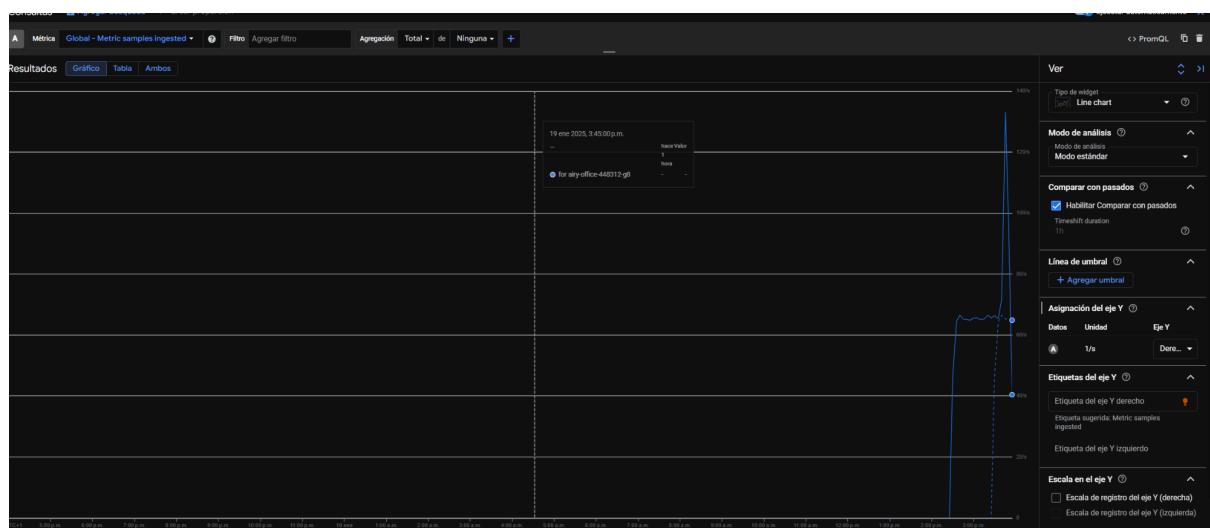
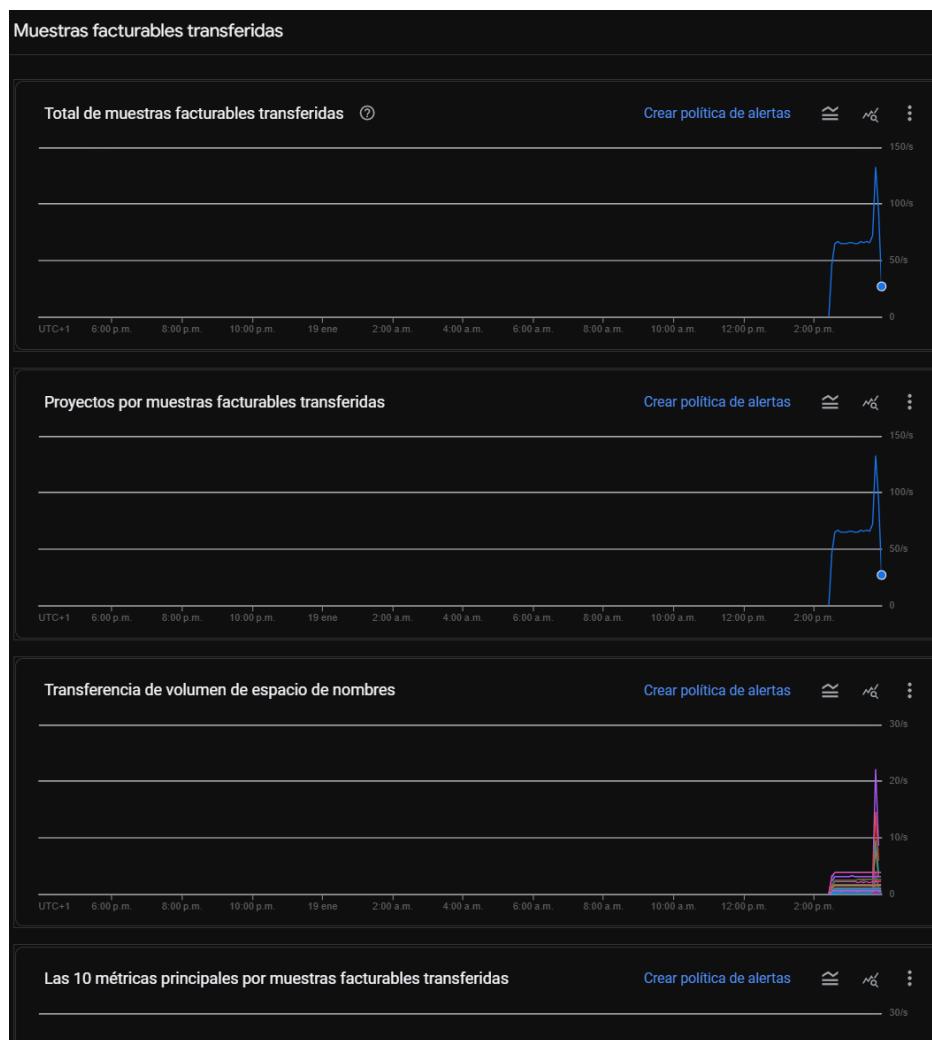
11º. En el panel podemos ver métricas como los logs que contiene el clúster.



12º. En el apartado de **administración de métricas** podemos ver una cantidad enorme de métricas que te vienen por defecto. En mi caso mostraré primero la facturación en bytes, transferidos o los errores.

The screenshot shows two main sections of the Grafana Metrics Explorer interface:

- Métricas (Metrics) View:** This view displays various metrics from different sources. Metrics are grouped by source (e.g., container, kubelet, pod, etc.) and include details like metric name, type (counter, gauge), and current value (N/A). A specific metric, "container_fs_reads_total/counter", is highlighted with a yellow box. The top right of this section shows summary statistics: 2 metrics active, 75 metrics factured by samples, 8B bytes facturable transferred, 363.1m samples factured transferred, and 0.01% errors.
- Consultas (Queries) View:** This view shows a histogram titled "Global - Metric bytes ingested". The x-axis represents time in UTC+1, and the y-axis represents the count of bytes ingested. The histogram shows a single large blue bar extending from the origin to approximately 3.00 GB, indicating a massive volume of data being processed.



13º. También hay un apartado para errores.

Si nos metemos en uno de los errores podemos ver una métrica y una información acerca de este.

14º. También podemos ver una serie de registros que está monitorizando.

15º. Antes puse ciertas capturas acerca de los paneles en cuanto a la facturación, bytes ...

Ahora mostraré una de **GKE** que viene por defecto.

The screenshot shows the Google Cloud Observability Monitoring interface. On the left, there's a sidebar with various monitoring categories like Metrics Explorer, Log Explorer, and Alerting. The 'Panels' tab is currently selected. In the main area, there's a table listing 'Todos Panoramas' (All Panels) with columns for Nombre (Name), Type, Etiquetas (Labels), and Acciones (Actions). One panel, 'GKE', is highlighted with a yellow box. The 'GKE' panel has several sub-options listed under it, such as 'GKE Active/Idle Clusters', 'GKE Cluster Monitoring', and 'GKE Compute Resources - Cluster View'. The 'GKE' panel itself is also highlighted with a yellow box.

Cuando nos metamos dentro de este, podemos ver información acerca de nodos, ct's...

The screenshot shows the GKE cluster monitoring interface. It's divided into four main sections: Clusters, Namespaces, Nodes, and Workloads. Each section has a table with columns for Nombre (Name), Alertas (Alerts), Etiquetas (Labels), Reinicios de contenedor (Container Restarts), Registros de errores (Error Logs), Uso de CPU (CPU Usage), Uso de memoria (Memory Usage), and Uso de disco (Disk Usage). The 'Clusters' section shows one cluster named 'cluster-1'. The 'Namespaces' section shows several namespaces like 'default', 'gke-managed-cm', 'gke-managed-system', 'gmp-public', and 'gmp-system'. The 'Nodes' section shows multiple nodes within the 'cluster-1' namespace. The 'Workloads' section shows various workloads like 'gke-cluster-1-default...', 'rule-evaluator', and 'strimzi-cluster-operator'. Each row in the tables provides specific usage statistics for each resource type.

Objetos Service de Kubernetes							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
cassandra-p20-cass...	0	Clúster: cluster-1 +3	0	0	44.86% de 1.5 CPU	8.34% de 12 GB	0.02% de 26.52 GB
kubernetes	0	Clúster: cluster-1 +3	0	0	0 CPU	0 B	0 B
alarmmanager	0	Clúster: cluster-1 +3	0	0	0 CPU	0 B	0 B
grpnp-operator	0	Clúster: cluster-1 +3	0	0	251.56% de 0 CPU	62.89% de 15.26 ...	0% de 96.12 GB
default-http-backend	0	Ubicación: un-central-1 Proyecto: any-office-44312-g8 Espacio de nombres: grpnp-system	0	0	0.62% de 0.01 CPU	8.79% de 20 MB	0 B

Pods							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
cassandra-p20-cass...	0	Clúster: cluster-1 +3	0	0	46.09% de 0.5 CPU	8.06% de 4 GB	0.02% de 8.84 GB
cassandra-p20-cass...	0	Clúster: cluster-1 +3	0	0	48.68% de 0.5 CPU	7.88% de 4 GB	0.02% de 8.84 GB
cassandra-p20-cass...	0	Clúster: cluster-1 +3	0	0	39.82% de 0.5 CPU	9.07% de 4 GB	0.02% de 8.84 GB
cassandra-p20-depl...	0	Clúster: cluster-1 +3	0	0	0 CPU	0 B	0 B
kube-state-metrics-0	0	Clúster: cluster-1 +3	0	0	2.96% de 0.11 CPU	11.58% de 130 MB	0% de 94.77 GB

Contenedores							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
cassandra	0	Clúster: cluster-1 +4	0	0	46.09% de 0.5 CPU	8.06% de 4 GB	
cassandra	0	Clúster: cluster-1 +4	0	0	48.68% de 0.5 CPU	7.88% de 4 GB	
cassandra	0	Clúster: cluster-1 +4	0	0	39.82% de 0.5 CPU	9.07% de 4 GB	
deployer	0	Clúster: cluster-1 +4	0	0	0 CPU	0 B	
ksm-metrics-collector	0	Clúster: cluster-1 +4	0	0	5.67% de 0.01 CPU	18.18% de 30 MB	

Como se puede apreciar aquí abajo hay 76 contenedores que se crean por defecto al levantar el clúster

Contenedores							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
cassandra	0	Clúster: cluster-1 +4	0	0	46.09% de 0.5 CPU	8.06% de 4 GB	
cassandra	0	Clúster: cluster-1 +4	0	0	48.68% de 0.5 CPU	7.88% de 4 GB	
cassandra	0	Clúster: cluster-1 +4	0	0	39.82% de 0.5 CPU	9.07% de 4 GB	
deployer	0	Clúster: cluster-1 +4	0	0	0 CPU	0 B	
ksm-metrics-collector	0	Clúster: cluster-1 +4	0	0	5.67% de 0.01 CPU	18.18% de 30 MB	

Entre todos esos podemos ver **grafana, prometheus, fluentbit....** (Que sirven para monitorizar el clúster de Cassandra, esta arquitectura viene por defecto ya creado)

Contenedores							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
kube-state-metrics	0	Clúster: cluster-1 +4	0	0	0.25% de 0.1 CPU	4.97% de 100 MB	
config-reloader	0	Clúster: cluster-1 +4	0	0	11.41% de 0 CPU	65.81% de 3.81 M...	
prometheus	0	Clúster: cluster-1 +4	0	0	113.58% de 0 CPU	69.06% de 30.52 ...	
config-reloader	0	Clúster: cluster-1 +4	0	0	16.63% de 0 CPU	65.15% de 3.81 M...	
prometheus	0	Clúster: cluster-1 +4	0	0	108.69% de 0 CPU	62.13% de 30.52 ...	

Contenedores							
Nombre	Alertas	Etiquetas	Reinicios de contenedor	Registros de errores	Uso de CPU	Uso de memoria	Uso de disco
strimzi-cluster-oper...	0	Clúster: cluster-1 +4	0	1	4.3% de 0.2 CPU	27.03% de 384 MB	
strimzi-cluster-operator	0	Clúster: cluster-1 +4	0	6	9.8% de 0 CPU	5.88% de 100 MB	
prometheus-to-sd-ex...	0	Clúster: cluster-1 +4	0	0	0 CPU	9.47 MB	
fluentbit	0	Clúster: cluster-1 +4	0	0	10.73% de 0.05 C...	7.45% de 100 MB	
fluentbit-gke	0	Clúster: cluster-1 +4	0	0	0.55% de 0.05 CPU	6.21% de 100 MB	

Además si queremos crear un filtro para monitorizar podemos agregarlo dándonos las siguientes optionalidades

Añadir filtro

Select a GKE resource

Search GKE resources

Clústeres	1 total >
Espacios de nombres	9 total >
Nodos	4 total >
Cargas de trabajo	39 total >
Objetos Service de Kubernetes	7 total >
Pods	38 total >

Espacio de nombres: gmp-system, Clúster: cluster-1, Ubicación: eu-central-1

- collector-s828s**
- Espacio de nombres: gmp-system, Clúster: cluster-1, Ubicación: eu-central-1**
- gmp-operator-5d77846874-wvxn9**
- Espacio de nombres: gmp-system, Clúster: cluster-1, Ubicación: eu-central-1**
- strimzi-cluster-operator-66b5ff8bbb-sng8v**
- Espacio de nombres: kafka, Clúster: cluster-1, Ubicación: eu-central-1**
- event-exporter-gke-79cd469d79-4sp9t**
- Espacio de nombres: kube-system, Clúster: cluster-1, Ubicación: eu-central-1**
- fluentbit-gke-9j97p**
- Espacio de nombres: kube-system, Clúster: cluster-1, Ubicación: eu-central-1**
- fluentbit-gke-bgpxd**

Restablecer X

16º. También hay una serie de integraciones con las que puedes unir tu clúster, yo intenté unir otra herramienta de monitorización por ver si podía implementarla y no conseguí hacerlo.

Estado de prueba gratuita. Creado hace 1287 días y 97 días restantes. Activa tu cuenta completa para obtener acceso ilimitado a todas las funciones de Google Cloud. Usa los créditos restantes y paga solo por lo que usas.

Google Cloud My First Project Buscar (/) recursos, documentos, productos y más

Observabilidad Monitoring

Comienza a usar las integraciones

Las integraciones reúnen métricas, registros, paneles y alertas a fin de brindarte acceso rápido a datos enriquecidos para la pila de tu aplicación. Debido a que está creada en una base de código abierto, puedes personalizar los datos según tus necesidades.

Integraciones

Filtros rápidos Todos Filtrar Integraciones

VM con agente de operaciones	Clústeres con Prometheus administrado
0 sin Agente de operaciones. Ver VMs	1 sin Prometheus administrado. Ver clústeres

Integraciones

- Active Directory Domain Services (AD DS)**
- Aerospike**
- Anthos**
- Apache ActiveMQ**
- Apache Airflow**

Configurar

- Integraciones** (selected)
- Métricas basadas en reglas
- Entrador de registros
- Almacenamiento de registros
- Administración de métricas
- My First Project
- Notas de versión

Todos estos son los apartados que te permiten integrar dentro de tu clúster; si clickas en uno de estos te muestra una serie de aplicaciones que se pueden usar para poder realizar la tarea que quieras.

Cumpla con las prácticas recomendadas

Conoce las prácticas recomendadas y las recomendaciones de Google sobre cómo ejecutar los clústeres de grado de producción en GKE.

INTEGRACIÓN ASISTIDA DE GKE Completado: 13 %

Pasos de integración recomendados

- Preparation OPEN
- Environment OPEN
- Cluster config OPEN
- Security OPEN
- Networking OPEN
- Multi-tenancy OPEN
- Monitoring OPEN
- Maintenance OPEN

CONCLUSIÓN

En conclusión, Cassandra se destaca como una base de datos distribuida y altamente escalable, ideal para gestionar grandes volúmenes de datos con disponibilidad y rendimiento garantizados. A lo largo de este análisis, he comprendido su arquitectura robusta, su enfoque en la tolerancia a fallos y las herramientas que complementan su uso, como Prometheus, Fluent Bit y Grafana, que facilitan la monitorización y la gestión eficiente. Además, he explorado su implementación práctica, desde la instalación hasta la configuración de seguridad y optimización. Este conocimiento no solo refuerza su importancia en proyectos de datos críticos, sino que también me inspira a aplicar esta tecnología en soluciones innovadoras que requieran resiliencia y escalabilidad.

Puedo destacar muchas herramientas usadas pero yo me quedo sobre todo con **datadog** y **AxonOPS** que son 2 herramientas monstruosas y que no conocía.

WEBGRAFÍA

INFORMACIÓN ACERCA DE CASSANDRA

<https://paradigmadigital.com/dev/cassandra-la-dama-de-las-bases-de-datos-nosql/>

<https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/apache-cassandra/>

<https://openwebinars.net/blog/que-es-apache-cassandra/>

COMO INSTALAR Y CONFIGURACIONES CASSANDRA

<https://docs.vultr.com/how-to-install-apache-cassandra-on-debian-12>

CONEXIONES REMOTAS CASSANDRA

➡ Como Habilitar Conexión Remota en Apache Cassandra y uso de RazorSQL

PASOS PARA INSTALAR E IMPLEMENTAR DATADOG

<https://www.programador-web.com/2024/11/01/guia-completa-para-principiantes-monitoreo-y-gestion-con-datadog/>