UNIVERSIDAD DE COSTA RICA

FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS DE LA COMPUTACIÓN E INFORMÁTICA

Cassandra: Una base de datos distribuida ideal para el modelado del Big Data.

AUTORES - EQUIPO "Thunderbolts"

Joshua Ramírez Alfaro B76136
Mario Viquez González B78443
Daniel Sancho Varela B66676

FECHA - 28 de junio, 2022

RESUMEN - ABSTRACT

El presente artículo corresponde a una investigación de tipo descriptiva sobre la herramienta Apache Cassandra en torno al análisis de grandes volúmenes de datos. Apache Cassandra es bastante diferente de un sistema de administración de bases de datos relacionales (RDBMS) típico. Apache Cassandra es una base de datos NoSQL de código abierto enormemente escalable, la cual es perfecta para administrar grandes cantidades de datos estructurados, semiestructurados y no estructurados en múltiples centros de datos y la nube. En las diferentes secciones del documento, se detallan aspectos interesantes de la herramienta como sus orígenes, evolución, funcionamiento, ventajas y desventajas, y una comparación con una base de datos de la misma categoría como lo es MongoDB. Para finalizar, se muestra un ejemplo pequeño de una estructura base realizada en Cassandra para observar algunas de sus propiedades más importantes en funcionamiento.

PALABRAS CLAVES - KEYWORDS

Apache Cassandra* NoSQL* Escalabilidad *Nube* MongoDB* RDBMS*

INTRODUCCIÓN

En los últimos años, la tecnología se ha convertido en un factor determinante en el realce del valor de los datos. Gracias a ella, ahora se accede a información a la que antes no se tenía alcance y se crean sistemas que aumentan la capacidad de procesamiento y de almacenamiento, permitiendo obtener resultados casi en tiempo real de millones de registros, sin embargo, no todos los sistemas se encuentran capacitados para almacenar mucha información. A la hora de gestionar volúmenes de datos muy grandes, de varios terabytes o incluso petabytes, los sistemas de bases de datos clásicos están desbordados. Para estos casos, se necesitan aplicaciones de **Big Data**, que sean fácilmente escalables, ya que la mayoría de las veces, es difícil estimar de antemano el volúmen real de los datos . Uno de los ejemplos más modernos y más populares de estas aplicaciones corresponde a Apache Cassandra. Esta herramienta provee de funcionalidades y características muy interesantes, las cuales vamos a detallar a continuación mediante la contestación de diferentes preguntas.

HISTORIA, CREACIÓN Y ORÍGENES

Su desarrollo comenzó en Facebook, donde se diseñó para impulsar la función de búsqueda en la bandeja de entrada, para ayudar a los usuarios de Facebook a buscar datos en sus bandejas de entrada más fácilmente. En 2008, se publicó como proyecto de código abierto; en febrero de 2010, se convirtió en un proyecto de primer nivel de la Apache Software Foundation. Está inspirado e influenciado por los trabajos de 2007 de Amazon Dynamo y 2006 de Google

Bigtable. Actualmente, es mantenido y desarrollado por Datastax.
El 17 de febrero de 2010 se graduó como proyecto de alto nivel.

Las versiones posteriores a la graduación incluyen:

- 0.6, publicada el 12 de abril de 2010, añadió soporte para el almacenamiento en caché integrado y Apache Hadoop MapReduce
- 0.7, publicada el 8 de enero de 2011, añadió índices secundarios y cambios en el esquema en línea
- 0.8, publicada el 2 de junio de 2011, añadió el Lenguaje de Consulta Cassandra (CQL), memtables autoajustables y soporte para actualizaciones sin tiempo de inactividad
- 1.0, publicada el 17 de octubre de 2011, añadió compresión integrada, compactación nivelada y mejora del rendimiento de lectura
- 1.1, publicada el 23 de abril de 2012, añadió cachés autoajustables, aislamiento a nivel de filas y soporte para despliegues mixtos de ssd/disco giratorio
- 1.2, publicada el 2 de enero de 2013, añadió la agrupación en clústeres a través de nodos virtuales, la comunicación entre nodos, los lotes atómicos y el seguimiento de solicitudes
- 2.0, publicada el 4 de septiembre de 2013, añadió transacciones ligeras (basadas en el protocolo de consenso Paxos), disparadores, compactaciones mejoradas, soporte de paginación CQL, soporte de sentencias preparadas, soporte de alias de columnas SELECT.

Actualmente, la herramienta se ejecuta tanto en dispositivos con sistema operativo Windows, como con sistemas similares a UNIX y en preferencia en servidores Linux, siendo este último el que posee mejor rendimiento. Si se desea obtener más información acerca de su instalación, puede ingresar al repositorio de control de versiones donde tendrá acceso a una guía completa para ambos sistemas operativos.

¿POR QUÉ UTILIZAR APACHE CASSANDRA PARA BIG DATA?

Su importancia en el Big Data se resume en que gestiona los datos en forma de clusters interconectados a miles de nodos repartidos por los centros de datos. Se le conoce como una "base de datos orientada a columnas" en NoSQL, en la que la información se almacena columna por columna, en contraste con el enfoque basado en filas de los sistemas de bases de datos tradicionales. Debido a esto tiene menos operaciones de E/S para poder almacenar la información.

Utilizar Apache Cassandra en el ámbito de Big Data se ha vuelto muy común debido a la buena escalabilidad en combinación con una probabilidad de fallo muy baja, dos condiciones indispensables para las aplicaciones de big data. De igual manera, esta herramienta posee una curva de aprendizaje relativamente corta, ya que Cassandra utiliza su propio lenguaje llamado CQL (Cassandra Query Language), el cual corresponde al propio lenguaje SQL, pero sin sus features más avanzados. La principal diferencia con SQL es que Cassandra no admite combinaciones ni subconsultas. En cambio, Cassandra enfatiza la desnormalización a través de características de CQL como colecciones y agrupaciones especificadas en el nivel de esquema. Esta simpleza hace que sea muy fácil manejar la herramienta de Cassandra en un periodo de tiempo más corto. Por último, esta fue dotada de un increíble rendimiento precisamente para manejar muchos servidores básicos, proporcionando alta disponibilidad sin un punto único de fallo; y no es casualidad que Cassandra triunfe actualmente entre los proveedores de grandes redes sociales que requieren la interconexión de grandes volúmenes de datos entre los usuarios.

Se ha utilizado principalmente en aplicaciones de grandes datos que utilizan

información en tiempo real, como los procedentes de componentes de sensores o de sitios web de redes sociales. Algunos de los clientes más conocidos son Twitter, Instagram, Spotify, Netflix así como el sitio web Digg o el portal de noticias sociales Reddit.

Figura 1. Ejemplos de Empresas que usan Apache Cassandra

Además, Cassandra tiene una arquitectura descentralizada, lo que significa que los módulos de funciones como el particionamiento de datos, la replicación, el escalado y la gestión de fallos están presentes por separado y trabajan en conjunto, significando que cualquier nodo puede encargarse de cualquier operación de procesamiento de datos.

DESCRIPCIÓN, FUNCIONAMIENTO, CARACTERÍSTICAS Y ARQUITECTURA DE CASSANDRA

Primeramente, partamos por responder la siguiente pregunta: **¿Qué es Apache Cassandra?**

Apache Cassandra se trata de un software NoSQL distribuido y basado en un modelo de almacenamiento de «clave-valor», de código abierto que está escrito en Java, que permite grandes volúmenes de datos en forma distribuida [4]. Es una solución open source distribuida, diseñada para almacenar grandes cantidades de datos proporcionando una alta disponibilidad sin un único punto de fallo. Esta herramienta es muy diferente de las bases de datos convencionales, con características que la hacen realmente **única**, entre ellas podemos mencionar las siguientes:

- Posee un servicio altamente disponible (Permite su uso a cualquier hora del día) y sin un punto único de fallo.
- Verdadera consistencia en acceso y disponibilidad.
- Es capaz de escribir de manera rápida grandes cantidades de datos sin afectar la efectividad de las lecturas.
- Y además tiene precisión y velocidad sin dependencia en el volumen de los datos. (No importa si se manejan cantidades pequeñas o grandes de datos, la precisión y velocidad no se va a ver afectada).
- No tiene un único punto de falla

Ahora que se conoce las características que hacen única a la herramienta es momento de contestar a lo siguiente:

¿Cuál es la arquitectura utilizada en la herramienta?

Primeramente, hay que hacer un repaso sobre los componentes de la misma, estos corresponden a los siguientes [13]:

- **Nodo** : es el lugar donde se almacenan los datos.

- **Centro de datos** : es una colección de nodos relacionados.

- **Clúster** : un clúster es un componente que contiene uno o más centros de datos.

- **Registro de confirmación (Commit log):** el registro de confirmación es un mecanismo de recuperación de fallas en

Cassandra. Cada operación de escritura se escribe en el registro de confirmación. Guardan la información de cada consultaque realicemos. Si tenemos una caída o error, podemos hacer una recuperación mediante los logs

- **Mem-table :** Una mem-table es una estructura de datos residente en la memoria. Después del registro de confirmación, los datos se escribirán en la tabla mem. A veces, para una familia de una sola columna, habrá varias tablas de memoria.

- **SSTable :** Es un archivo de disco al que se descargan los datos almacenados en las tablas de memoria, y son ordenados para un acceso rápido y un almacenamiento en el disco en un conjunto de archivos persistente, ordenado e inmutable.

- **Filtro Bloom**: Son más que algoritmos rápidos y no deterministas para probar si un elemento es miembro de un conjunto. Es un tipo especial de caché. Se accede a los filtros Bloom después de cada consulta. Con ella podemos determinar la pertenencia de un elemento a un conjunto.

Algunas de las **características** más importantes que nos proporciona esta herramienta son las siguientes [11]:

Es distribuida: Esto quiere decir que la información está repartida a lo largo de los nodos del clúster. Esta característica resulta muy útil para clusters. Además ofrece alta disponibilidad, de manera que si alguno de los nodos se cae el servicio no se degradará. Asimismo, esta realiza una comunicación de tipo peer-to-peer(p2p), muy diferente a las bases de datos convencionales que usan el protocolo de Single Point of Failure. Este protocolo es muy importante por que a diferencia de Single Point, donde tenemos un único servidor que realiza las transacciones, si este llega a fallar, va a afectar a todos los nodos, ya que no habría comunicación entre ellas. El protocolo peer to peer hace que si uno de los nodos se cae, dado que existe réplica, se comienza a utilizar el siguiente nodo. No sigue patrones maestro-esclavo como otros sistemas de almacenamiento, de esta manera cualquiera de los nodos puede tomar el rol de coordinador de una query

Escala linealmente: Esto quiere decir que el rendimiento varía de forma lineal respecto al número de nodos que añadamos. Por ejemplo, si con 2 nodos soportamos 100.000 operaciones por segundo, con 4 nodos soportamos 200.000. Esto da mucha predictibilidad a nuestros sistemas.

Escala de forma horizontal: Lo que quiere decir que podemos escalar nuestro sistema añadiendo nuevos nodos basados en hardware commodity de bajo coste.

Almacenamiento de datos flexible : Cassandra acomoda todos los formatos de datos posibles, incluyendo: estructurados, semi-estructurados y no estructurados. Puede acomodar dinámicamente los cambios en sus estructuras de datos de acuerdo con sus necesidades.

Fácil distribución de datos: Cassandra proporciona la flexibilidad de distribuir los datos donde usted quiera replicando los datos a través de múltiples centros de datos.

Escritura rápida: Cassandra fue diseñada para funcionar con hardware barato. Realiza una escritura increíblemente rápida y puede almacenar muchos terabytes de conocimiento, sin sacrificar la eficiencia de lectura.

Los sistemas como Cassandra están diseñados para estos retos y buscan los siguientes objetivos de diseño:

- Replicación completa de la base de datos multimaster. La

replicación multimaster es un método de replicación de bases de datos que permite que los datos sean almacenados por un grupo de ordenadores y actualizados por cualquier miembro del grupo. Todos los miembros responden a las consultas de datos de los clientes. El sistema de replicación multimaster se encarga de propagar las modificaciones de datos realizadas por cada miembro al resto del grupo y de resolver los conflictos que puedan surgir entre los cambios concurrentes realizados por diferentes miembros

- Disponibilidad global con baja latencia.
- Escalado en hardware básico.
- Aumento lineal del rendimiento con cada procesador adicional.
- Equilibrio de carga en línea y crecimiento del clúster.
- Consultas orientadas a la clave con partición.
- Esquema flexible.

Cassandra trabaja con nodos en un cluster como se mencionaba anteriormente en el aspecto de multimaster. Normalmente, una solicitud de lectura/escritura de una clave se dirige a cualquier nodo del clúster de Cassandra. El nodo determina entonces las réplicas para esta clave en particular. Para las escrituras, el sistema redirige las peticiones a las réplicas y espera a que un quórum de réplicas reconozca la finalización de las escrituras. Para las lecturas, basándose en las garantías de consistencia requeridas por el cliente, el sistema dirige las solicitudes a la réplica más cercana o a todas las réplicas y espera un quórum de respuestas.

Gracias a estas características tan interesantes, Apache Cassandra nos puede brindar una serie de **funciones principales** como por ejemplo:

Permite la replicación en varios data center: Siendo cada data center un anillo de máquinas Cassandra, ya que permite que el anillo 1 replique sus datos en el anillo 2 [13]. Los datos se distribuyen a través del clúster. Todos los nodos del cluster tienen un mismo rol,por lo que todo nodo puede dar servicio a cualquier solicitud.

Es tolerante a fallos: Gracias a que posee la replicación de datos, es decir, los datos cuando son escritos en un nodo se replican en otros nodos, por lo que si uno de estos nodos cae, no pasa nada porque el dato está replicado en otros dos. Los datos se replican a múltiples nodos cada vez que son agregados. Los nodos que presentan fallos pueden recuperarse sin necesidad de detener o inactivar la aplicación. Para garantizar una baja probabilidad de fallo y el restablecimiento de los datos en caso de emergencia, Apache Cassandra cuenta con un sistema de replicación adecuadamente configurado. La tolerancia de error se minimiza porque los datos se replican automáticamente entre los nodos. Los nodos averiados se pueden sustituir muy fácilmente. El sistema permanece disponible para consultas en todo momento [13].

¿Cómo funciona el modelado de datos en Apache Cassandra?

En esta sección se detalla cómo funcionan los datos a la hora de ser almacenados. El modelo de datos de Cassandra es significativamente diferente de lo que normalmente vemos en un RDBMS

Cassandra proporciona un sistema de modelado de datos que incluye los siguientes puntos clave:

- Clave de partición

  Cassandra es una base de datos distribuida en la que los datos se particionan y almacenan a través de múltiples nodos dentro de un clúster.

La clave de partición se compone de uno o más campos de datos y es utilizada para generar un token a través de hashing que permite distribuir los datos de manera uniforme en un clúster.

- Clave de clustering
  Una clave de agrupación se compone de uno o más campos y ayuda a agrupar filas con la misma clave de partición y a almacenarlas en orden.

  Digamos que estamos almacenando datos de series temporales en Cassandra y queremos recuperar los datos en orden cronológico. Una clave de agrupación que incluya campos de datos de series temporales será muy útil para la recuperación eficiente de los datos para este caso de uso.

  Nota: La combinación de la clave de partición y la clave de clustering constituye la clave primaria e identifica de forma única cualquier registro en el cluster de Cassandra.

- Pautas en torno a los patrones de consulta Identificación de los patrones de consulta y asegurarnos de que se adhieren a las siguientes directrices:

1. Cada consulta debe obtener datos de una sola partición.
2. Debemos controlar la cantidad de datos que se almacenan en una partición, ya que Cassandra tiene límites en cuanto al número de columnas que se pueden almacenar en una sola partición.
3. Está bien desnormalizar y duplicar los datos para soportar diferentes tipos de patrones de consulta sobre los mismos datos.

¿Qué escenarios es apropiado utilizar la herramienta?. ¿Cuándo es apropiado usar Apache Cassandra?

Una aplicación ideal para el uso de Cassandra tiene las siguientes características:

- Las escrituras superan a las lecturas por un gran margen.
- Los datos se actualizan raramente y cuando se hacen actualizaciones son idempotentes (propiedad para realizar una acción determinada varias veces y aun así conseguir el mismo resultado que se obtendría si se realiza una sola vez).
- El acceso de lectura es por una clave primaria conocida.
- Los datos se pueden particionar mediante una clave que permite repartir la base de datos entre varios nodos.
- No es necesario realizar uniones o agregados.

Ejemplos:

- Registro de transacciones: Compras, puntuaciones de exámenes, películas vistas y última ubicación de la película.
- Seguimiento de prácticamente todo, incluyendo el estado de los pedidos, los paquetes, etc.
- Almacenamiento de datos de seguimiento de la salud.

COMPARACIÓN DE CASSANDRA CON MONGODB

MongoDB es una base de datos NoSQL de código abierto. Esta base de datos almacena datos de forma similar a JSON, es decir, como pares clave-valor en un documento. Cada documento se considera parte de una colección. MongoDB es una base de datos NoSQL que ofrece almacenamiento distribuido, escalado horizontal y alta disponibilidad.

Con un poco de contexto del funcionamiento de MongoDB se pueden ahora mencionar diferencias significativas entre ambas herramientas:

1. Apache Cassandra implementa una arquitectura de almacenamiento columnar y almacena los datos en forma de filas y columnas tradicionales. Cada columna tiene un tipo de datos específico que puede almacenar y que debe ser especificado en el momento de la creación de la tabla. Por otro lado, MongoDB almacena los datos en un formato similar a JSON en documentos que se almacenan en una colección. Esto significa que la estructura de los datos a puede cambiarse para cada registro y no requiere que esté en un formato predefinido. El almacenamiento en un formato similar a JSON también permite anidar los datos para que el registro sea más provechoso.

2. MongoDB tiene un único nodo maestro que controla varios nodos esclavos. Si el nodo maestro falla o se deshabilita, se inicia un proceso de elección automático donde se elige un nodo esclavo para tomar el campo del maestro. Este proceso puede tardar hasta uno o más minutos en completarse por lo que no se puede asegurar el 100% de disponibilidad. Por otro lado, Apache Cassandra tiene múltiples nodos maestros dentro de un cluster. Esto significa que si uno de los nodos maestros se cae, no hay tiempo de inactividad ya que otros que sí están activos pueden manejar las solicitudes entrantes. Entonces esta arquitectura es poderosa ya que asegura el 100% de disponibilidad en comparación a MongoDB.

3. Las bases de datos distribuidas sólo permiten que el nodo maestro realice operaciones de escritura y los nodos esclavos sólo realicen operaciones de lectura. Dado que MongoDB tiene un único Nodo Maestro, sólo puede realizar una operación de escritura a la vez y, por lo tanto, puede considerarse limitada en términos de escalabilidad de escritura. Por otro lado, Apache Cassandra, que tiene múltiples nodos maestros, puede coordinar múltiples operaciones de escritura al mismo tiempo. Por lo tanto , Apache Cassandra tiene una ventaja sustancial en términos de escalabilidad de escritura.

CONCLUSIÓN

Cassandra se presenta como una gran solución a la problemática del manejo de grandes bancos de datos, aun cuando la forma de modelar dista bastante de el modelo clásico relacional que todos conocemos, además, las herramientas para administrar nuestros Keyspaces, cluster y nodos nos ayudan a mantener un orden al momento de desarrollar y utilizar nuestras aplicaciones. El lenguaje CQL nos provee funciones bastante parecidas a lo que estábamos acostumbrados con SQL, haciendo más fácil el acostumbrarse a utilizar las diversas funcionalidades de esta tecnología.

RETOS ENCONTRADOS Y SOLUCIONES APLICADAS

Para esta entrega se nos presentaron algunas dificultades, las cuales fueron las siguientes:

- La herramienta Apache Cassandra es poco intuitiva de utilizar en sistemas operativos Windows. Muchos de los materiales, tutoriales y guías estaban enfocados fuertemente en sistemas de Linux. Dado que nuestro grupo es poseedor de sistemas con Windows en su mayoría, tuvimos inconvenientes. Ante esta dificultad, pensamos que la solución sería el

virtualizar una máquina local de Linux desde nuestros computadores (Usando Oracle VirtualBox). Sin embargo, su capacidad limitada de recursos no permitía un adecuado progreso de la investigación. Adicionalmente, la forma en la que arrancamos una Base de Datos de Cassandra fue mediante la herramienta Docker, y para crear el cluster, necesitábamos que un nodo sea alojado por un contenedor, este proceso consume mucha memoria RAM (Concretamente 4 GB RAM por contenedor). Dado la limitante de recursos que teníamos en su mayoría los miembros del grupo, nos vimos en la necesidad de realizar el ejemplo práctico en la mejor máquina que teníamos (Una con 16 GB RAM de capacidad). Como es una máquina de escritorio, no es posible desplazarla a la clase, por lo que se grabó un video de su funcionamiento para que este sea mostrado al público.

- Aplicar la comparación de MongoDB con la herramienta Cassandra desde un punto de vista más práctico no fue posible, debido a que el grupo se enfocó en lidiar con las dificultades técnicas que se nos presentaron con Apache Cassandra. Además, dado que la herramienta de interés en este caso es Cassandra, consideramos que no era viable enfocar parte del tiempo en instalar y ver como funciona en detalle MongoDB. ya que excede los objetivos planteados desde el inicio. Por lo que se opta por una comparativa más teórica, enfocando sus principales virtudes entre las bases de datos mencionadas,

- El aprendizaje de la herramienta se complicó un poco por su funcionamiento y características. En sí, implicó una curva de aprendizaje elevada, al ser un tipo de base de datos totalmente nuevo para nosotros, hemos tenido que adaptarnos a su funcionalidad un tanto peculiar y trabajar con diversos tutoriales para tener un conocimiento más sólido de esta base de datos.

- Ante la incompatibilidad de Windows, hubo que utilizar otra de las computadoras al alcance para poder hacer que funcionara correctamente.

- Se realizó un tutorial para mitigar errores durante el funcionamiento de Cassandra en Windows 10 para ser específico, utilizando Python 2 y Java.

- Algunos de las fuentes o documentación oficial que se puede encontrar en internet, no es del todo esclarecedora, o bien presenta errores ortográficos y de redacción, limitando un poco el avance y el entendimiento de la herramienta, para ello se optó por la búsqueda de otras fuentes o videos de información que fueron más útiles para la investigación en desarrollo.

MATERIALES UTILIZADOS

Para la realización de este trabajo, y para efectos de esta entrega se utilizaron los siguientes materiales:

Primeramente, se utilizó **equipo de cómputo** óptimo que nos ayudará a documentar y a realizar las labores prácticas necesarias, cada integrante posee las siguientes máquinas con las que trabajará de manera individual:

Computador de escritorio con las siguientes características:
-16 GB de RAM ddr4.
-Tarjeta de video con 8GB.
-Procesador Intel Core i7 7700.

Computadora portátil con las siguientes características:

- 8 GB de RAM ddr4.
-Tarjeta de video con 4 GB.
-Procesador Intel Core i5 9300.

Computadora portátil con las siguientes características:
- 8 GB de RAM ddr4.
-Tarjeta de video con 6 GB.
-Procesador Intel Core i7 8750h.

A su vez, fue necesario que cada uno de los integrantes instale ciertas herramientas de software como lo puede ser la propia **base de datos distribuida Apache Cassandra**. Este proceso de la instalación de Cassandra involucra otros aspectos como lo es la plataforma **Datastax DevCenter**, ideal para manipular la base de datos Cassandra usando CQL 3. De igual manera, se necesita instalar la última versión de **Java**, esto debido a que Cassandra se ejecuta en la Máquina Virtual de Java (JVM). A su vez, si se desea utilizar las herramientas de gestión de nodos de Cassandra, es necesario tener instalado **Python**. La herramienta **Docker** fue de gran ayuda para crear el ejemplo práctico de una forma más sencilla e intuitiva. Su versatilidad y la forma en la que proporciona una manera estándar de ejecutar código lo convierte en un gran aliado para la base de datos NoSQL Apache Cassandra. Por último, se utilizaron **máquinas virtuales,** sin embargo, estas no fueron de gran utilidad por su limitada capacidad.

REFERENCIAS CONSULTADAS

[1]. Cassandra - Introducción — DATA SCIENCE. (2020). Recuperado el 16 de Mayo del 2022, de página web: https://datascience.eu/es/programacion/cassandra-introduccion-2/

[2]. Welcome to Apache Cassandra's documentation! Apache Cassandra Documentation. (2022). Apache Cassandra. https://cassandra.apache.org/doc/latest/

[3]. Cassandra Introduction: What is Apache Cassandra?. (2022). Recuperado el 6 de Junio del 2022, de página web: https://www.bmc.com/blogs/apache-cassandra-introduction/

[4]. What is Cassandra and why are big tech companies using it? | Ubuntu. (2022). Recuperado el 6 de Junio del 2022, de página web: https://ubuntu.com/blog/apache-cassandra-top-benefits

[5]. técnicas, C., & Cassandra, A. (2020). Apache Cassandra: gestión distribuida de grandes bases de datos. Recuperado el 6 de Junio del 2022, de página web: https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/apache-cassandra/

[6]. Cassandra: NoSQL, BigData, escalación y replicación. (2018). Recuperado el 6 de Junio del 2022, de página web: https://www.panel.es/bigdata-cassandra-escalacion-y-replicacion/

[7]. Apache Cassandra: Introducción - Aprender BIG DATA desde cero. (2019). Recuperado el 6 de Junio del 2022, de página web: https://aprenderbigdata.com/introduccion-apache-cassandra/

[8]. Top 5 reasons to use the Apache Cassandra Database. (2018). Recuperado el 6 de Junio del 2022, de página web: https://fedakv.medium.com/top-5-reasons-to-use-the-apache-cassandra-database-d541c6448557

[9]. Cassandra, la dama de las bases de datos NoSQL . (2022). Recuperado el 6 de Junio del 2022, de página web: https://www.paradigmadigital.com/dev/cassandra-la-dama-de-las-bases-de-datos-nosql/

[10]. Apache Cassandra | Apache Cassandra Documentation. (2022). Recuperado el 6 de Junio del 2022, de

página web: https://cassandra.apache.org/_/index.html

[11].Cassandra use Cases. (2022). Recuperado el 20 de Junio del 2022, de página web: https://blog.pythian.com/cassandra-use-cases/

[12].Usando Docker para construir un clúster Cassandra - programador clic. (2022). Retrieved 28 June 2022, from https://programmerclick.com/article/5822414869/

[13]. Cassandra - Data Model. (2022). Retrieved 28 June 2022, from https://www.tutorialspoint.com/cassandra/cassandra_data_model.htm