Calderón Sánchez, Mónica

Vlad Azpeitia, Álvaro

CASSANDRA DB



10/03/2020

Trabajo teórico para Modelos Avanzados de Bases de Datos

Índice

[INTRODUCCIÓN 2](#_Toc34752313)

[HISTORIA 2](#_Toc34752314)

[DEFINICIÓN Y CARACTERÍSTICAS 3](#_Toc34752315)

[ELEMENTOS DE UNA BASE DE DATOS CASSANDRA 4](#_Toc34752316)

[FUNCIONAMIENTO 4](#_Toc34752317)

[MODELOS DE DATOS QUE SOPORTA LA BASE DE DATOS 5](#_Toc34752318)

[CONSULTAS CQL 7](#_Toc34752319)

[EJEMPLO DE INSTALACIÓN 8](#_Toc34752320)

[PUESTA EN MARCHA 17](#_Toc34752321)

[ENUMERACIÓN Y EXPLICACIÓN DE CASOS DE USO PARA LOS QUE ESTÁ PENSADA 22](#_Toc34752322)

[ENUMERACIÓN DE LOS CASOS DE ÉXITO 23](#_Toc34752323)

[Digg 23](#_Toc34752324)

[Twitter 23](#_Toc34752325)

[CONCLUSIÓN 24](#_Toc34752326)

[BIBLIOGRAFÍA 24](#_Toc34752327)

# INTRODUCCIÓN

En el este trabajo que se desarrolla a continuación, se expondrán, tanto de manera teórica como práctica (mediante capturas de pantalla y ejemplos), los puntos enumerados anteriormente en el índice. Tratando de aclarar los conceptos que abarca cada uno de estos puntos, así como exponerlos adaptándonos a las herramientas que nos ofrece CassandraDB.

# HISTORIA

Fue hace más de 10 años, en 2008, cuando la gran empresa de Facebook se encuentra con la necesidad de desarrollar para su propio negocio, un sistema de almacenamiento, que permitiese atender búsquedas en tiempo real dentro de la misma aplicación.

Este gran reto fue aceptado por Avinash Lakshman (uno de los autores de Amazon’s Dynamo) y Prashant Malik quienes desarrollaron lo que hoy conocemos como CassandraDB. Tras crearlo, la propuesta fue lanzada como un proyecto de open source en el Google Code de 2008.

Más tarde, en marzo de 2009, el modelo de base de datos fue adquirido por la empresa Apache como proyecto de Apache Incubator. Después de esto, en 2010, CassandraDB se estableció como un proyecto top-level de la fundación Apache.

Actualmente, el mantenimiento y desarrollo está en manos de Datastax. La última versión lanzada y la única que tiene soporte actualmente es la 3.11.6 lanzada en 2017 y liberada en el 2019.

# DEFINICIÓN Y CARACTERÍSTICAS

Apache Cassandra es una base de datos no relacional (NoSQL) distribuida y basada en un modelo de almacenamiento clave-valor, de código abierto y escrita en Java. Asegura una alta escalabilidad y disponibilidad de los datos sin comprometer el rendimiento. Su soporte para la replicación en múltiples data centers es uno de los mejores del sector en cuanto a datos de latencia.

Que una base de datos sea no relacional, es decir, que sea NoSQL, significa que es una base de datos más flexibles a la hora de recoger datos desestructurados o semiestructurado. Estas bases de datos son estructuras que nos permiten almacenar información en aquellas situaciones en las que las bases de datos relacionales generan ciertos problemas debido principalmente a problemas de escalabilidad y rendimiento de las bases de datos relacionales donde se dan cita miles de usuarios concurrentes y con millones de consultas diarias. Además de lo comentado anteriormente, las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad–relación.

La característica de que una base de datos sea distribuida, implica que hay varias bases de datos, las cuales están lógicamente relacionadas, y se encuentran distribuidas en diferentes espacios lógicos y geográficos. Estos espacios están interconectados por una red de comunicaciones. Estas bases de datos cuentan con la capacidad de realizar procesamientos autónomos, lo que permite realizar operaciones locales o distribuidas.

Características:

* **Tolerancia a fallos**: los datos son automáticamente replicados a través de los nodos. Los nodos fallidos pueden reemplazarse sin pérdida de tiempo.
* **Rendimiento**: Cassandra supera en rendimiento a las alternativas más populares debido a su arquitectura.
* **Completamente distribuida:** cada máquina de Cassandra maneja una parte proporcional de cada actividad del sistema.
* **Motor de almacenamiento estructurado en forma de log:** un motor de estructura en forma de log evita la sobreescritura para hacer las actualizaciones de E/S secuenciales tanto en HDDs como SSDs. En HDDs porque la penalidad en la búsqueda es muy alta, en SSDs para evitar la amplificación de escritura y los fallos en el disco.
* **Estrecha integración con su motor de almacenamiento.**
* **Almacenamiento gestionado localmente.**
* **Descentralizada:** todos los nodos son replicas idénticas los unos de los otros.
* **Escalable:** Cassandra se ha empleado en despliegues de producción del orden de decenas de miles de nodos manejando decenas de petabytes de datos. *(Imagen 1)*

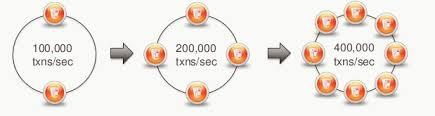


Imagen 1: gráfico sobre la escalabilidad.

* **Durabilidad:** Cassandra está pensada para aplicaciones que no pueden permitirse la pérdida de datos, incluso aunque un datacenter completo se caiga.
* **Alto control:** puedes elegir entre replicado asíncrono o síncrono.
* **Elasticidad:** la velocidad de lectura/escritura aumenta linealmente por cada nodo que se introduzca en la red, sin tiempo de espera o interrupciones.

# ELEMENTOS DE UNA BASE DE DATOS CASSANDRA

* **Commit log**: es el fichero donde se almacena la información sobre los cambios en los datos. Sirve para recuperar los datos en caso de fallo del sistema.
* **MemTable**: estructura de almacenamiento en memoria. Contiene los datos que aún no han sido introducidos en memoria.
* **SSTable**: es un fichero que almacena los datos escritos en disco. Cada fichero SSTable es inmutable una vez creado.
* **About internode communications (gossip)**: es el protocolo de comunicación p2p para descubrir y compartir información sobre la localización y estado de los nodos de un clúster de Cassandra.
* **Partitioner**: este componente determina cómo se distribuyen los datos entre los nodos.
* **Replica placement strategy**: define la estrategia a seguir para almacenar las copias de los datos en diferentes nodos, de forma que se aseguren la accesibilidad y la tolerancia a fallos.
* **Snitch**: define la topología que utilizan las estrategias de replicación para colocar las réplicas y dirigir las consultas de forma eficiente.

# FUNCIONAMIENTO

El proceso de escritura se realiza de la siguiente manera *(Imagen 2)*:

1. El usuario realiza una petición de escritura, esta se envía al MemTable, que almacena la petición de manera temporal en RAM.
2. Después se realiza la escritura en cada nodo y se escribe la operación en el commit log.
3. Cuando la MemTable se queda sin espacio, se limpian los datos.
4. Esos datos que se han limpiado se escriben en disco, en las SSTables, los cuales son inmutables y no pueden volver a tocarse.
5. Terminado el proceso, Cassandra intenta optimizar las SSTables eliminando aquellas que han quedado obsoletas porque han sido escritas de nuevo con datos actualizados en nuevas SSTables. Esto se conoce como compactación.

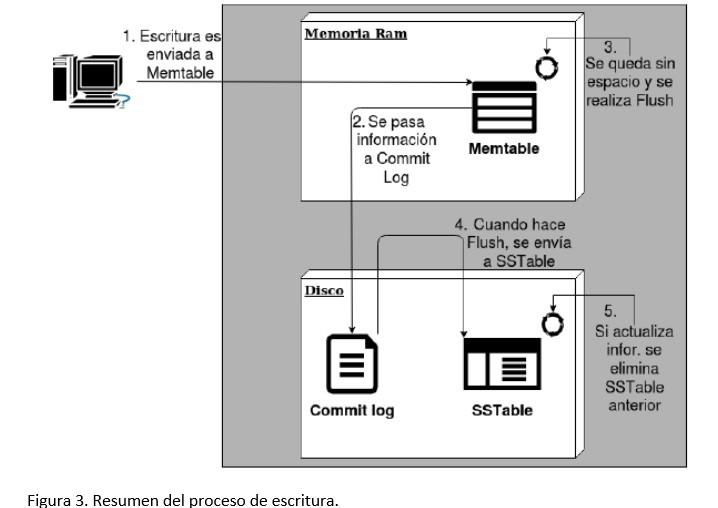


Imagen 2: resumen del proceso de escritura

El proceso de lectura se realiza de diferentes maneras dependiendo de la disposición de los datos. Si los datos se encuentran en la MemTable, se recuperarán de ahí de manera muy rápida. Si no se encuentra se busca en la SSTable. Dentro de estas dos maneras de obtener la información, existen tres tipos de lecturas:

* Lectura directa: el nodo coordinador contacta con un nodo que tenga la réplica requerida.
* Lectura digerida: esta lectura sirve para satisfacer el nivel de consistencia impuesto en el sistema. Se contacta con tantos nodos con la réplica requerida como dicte el nivel de consistencia y se comprueba la consistencia comparando los datos con los devueltos por el nodo contactado en lectura directa.
* Reparación de fallos en segundo plano: en el caso de encontrar inconsistencias en las réplicas, las que tengan un timestamp más antiguo son sobrescritas por la más actual.

En Cassandra, la administración de un claster se representa como un espacio circular o anillo. El anillo está dividido en rangos iguales al número de nodos, y cada nodo es responsable de uno o más rangos de los datos. Antes de que un nodo pueda formar parte del anillo, se le debe asignar un token. El token determina la posición del nodo en el anillo y el rango de datos del que es responsable. Los datos de la familia de columnas se dividen en los nodos según la clave de fila. Para determinar el nodo donde se encuentra la primera réplica de una fila el anillo se recorre en sentido horario hasta que se localice un nodo con un valor de token mayor que el de la tecla de fila. Cada nodo es responsable de su región del anillo y de la de su predecesor. El último nodo se considera el predecesor del primero, de ahí la estructura de anillo.

# MODELOS DE DATOS QUE SOPORTA LA BASE DE DATOS

Una base de datos Cassandra tiene una estructura totalmente diferente a una base de datos relacional. Algunos de los elementos tienen nombres similares a los de una base de datos relacional y algunos otros pueden tener similitudes, pero hay que tener muy en cuenta que Cassandra tiene una estructura completamente diferente a un modelo relacional y que esos elementos o similitudes no se corresponden con lo que representan en un modelo relacional

Elementos:

* **Columna:** es una estructura de tres campos que contiene: nombre de la columna, valor y timestamp. El nombre (clave) y el valor es lo que introduciría el usuario. El timestamp se genera de manera automática y su función es interna y oculta al usuario (sirve para resolver conflictos de consistencia entre datos como se ha mencionado anteriormente en el punto **Funcionamiento**) Un ejemplo en JSON sería el siguiente:

{

name: "userName",

value: "Dave Jones",

timestamp: 125555555

}

* **Super columna:** elemento compuesto de varias columnas.

{

name: "address",

value: {

city: {name: "city", value: "San Francisco", timestamp: 125555555},

street: {name: "street", value: "555 Union Street", timestamp: 125555555},

zip: {name: "zipcode", value: "94105", timestamp: 125555555},

}

}

Cassandra utiliza como clave el nombre de la columna:

city: {name: "city", …

* **Familia de columnas:** es un contenedor para una colección ordenada de columnas.

UserProfile = {

name: "user profile",

Dave Jones: {

email: {name: "email", value: "dave@email.com", timestamp: 125555555},

userName: {name: "userName", value: "Dave", timestamp: 125555555}

},

Paul Simon: {

email: {name: "email", value: "paul@email.com", timestamp: 125555555},

phone: {name: "phone", value: "4155551212", timestamp: 125555555},

userName: {name: "userName", value: "Paul", timestamp: 125555555}

}

}

* **Espacio de claves:** es una colección ordenada de familias de columnas. Es el equivalente a una base de datos en un modelo relacional.
* **Clúster:** conjunto de máquinas o nodos que conforman el sistema.

En Cassandra, los datos se guardan en columnas al igual que en un modelo relacional, pero el concepto de columna aquí es diferente al concepto de columna de un modelo relacional. En Cassandra, una columna no es más que un par clave-valor o clave-valores. Estas columnas se agrupan en familias de columnas dando lugar a algo similar pero no igual a una tabla de un modelo relacional. Es similar y no igual ya que en un modelo relacional se define primero el modelo y después se rellena con información. En un modelo relacional primero definimos una tabla y después rellenamos sus columnas. Aquí, rellenamos las columnas a la vez que las metemos en una familia de columnas. En las familias de columnas se pueden introducir nuevas columnas o eliminarlas a nuestro antojo, ya que, como hemos dicho antes, el concepto de columna aquí es totalmente diferente.

# CONSULTAS CQL

Cabe destacar, que CassandraDB usa un método de consultas que no es SQL, como veníamos acostumbrados a usar, sino que se trata de CQL (Contextual Query Language). Asimismo, Apache Cassandra ha querido llevar este tipo de consultas al extremo renombrando CQL como Cassandra Query Language.

CQL es un lenguaje formal para representar consultas a sistemas de recuperación de información. El diseño está orientado a que las consultas sean legibles y fáciles de entender por una persona, pero que siga manteniendo la expresividad de los lenguajes de consulta más complejos.

Las características que más lo diferencian de SQL son:

* **No hay cláusulas WHERE arbitrarias**: el predicado solo puede contener columnas especificadas en la clave primaria.
* **No existe el constructor JOIN:** no hay forma de unir datos entre familias de columnas.
* **No existe GROUP BY.**
* **No hay clausulas ORDER BY arbitrarias:** solo se puede aplicar a una columna de clúster.

Para empezar a ejecutar consultas, debemos empezar creando un keyspace (estructura similar a un esquema o una base de datos), escribiendo el siguiente CQL:

CREATE KEYSPACE animalkeyspace

WITH REPLICATION = { 'class' : 'SimpleStrategy' , 'replication\_factor' : 1 };

Después, debemos crear una familia de columnas con el comando CQL “use” de la siguiente forma:

use animalkeyspace;

Una vez hecho esto, podremos empezar a usar consultas más parecidas a SQL como CREATE TABLE o INSERT INTO.

CREATE TABLE Monkey (

identifier uuid,

species text,

nickname text,

population int,

PRIMARY KEY ((identifier), species));

INSERT INTO monkey (identifier, species, nickname, population)

VALUES ( 5132b130-ae79-11e4-ab27-0800200c9a66,

'Capuchin monkey', 'cute', 100000);

# EJEMPLO DE INSTALACIÓN

Para instalar CassandraDB en nuestro ordenador, lo primero que hemos hecho ha sido teclear: “CassandraDB” en la página de búsqueda de artículos científicos: [https://medium.com](https://medium.com/) y así nos hemos topado con un artículo llamado: “How to install CassandraDB” (El enlace a este artículo se encuentra en la bibliografía).

Siguiendo este tutorial y contrastándolo con una versión más reciente que podemos encontrar siguiendo el enlace de la página: [https://www.how2shout.com](https://www.how2shout.com/) que encontramos en la bibliografía, los pasos a seguir para instalar CassandraDB son los siguientes:

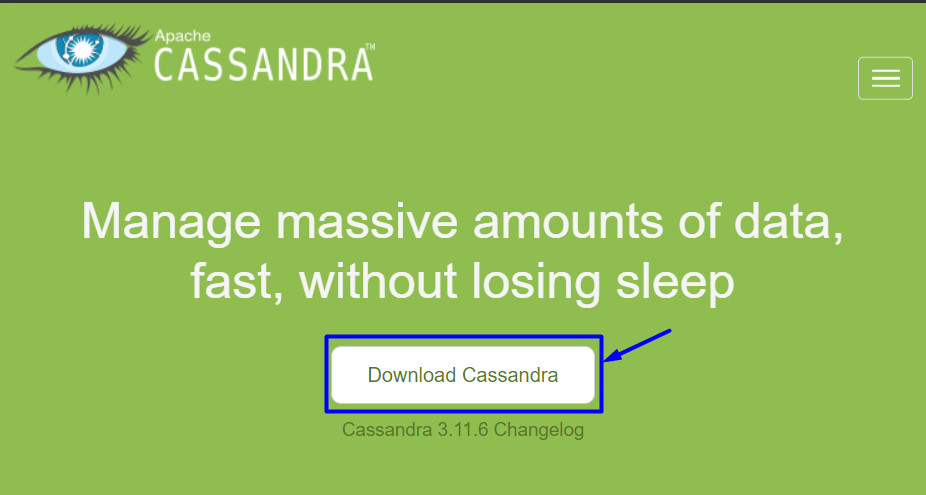
1. Ir a la página: “http://cassandra.apache.org/”
2. Darle clic al botón “Download Cassandra”. *(Imagen 3)*

Imagen 3: botón que nos redireccionará a una página de Apache para descargar el archivo comprimido que nos permitirá descargar CassandraDB.

1. Después de esto, seremos redirigidos a una página, donde podremos seleccionar la versión que deseamos instalar en nuestro ordenador.

Nosotros hemos elegido la versión más reciente a día de hoy, la versión 3.11.6, por lo que clicamos en el enlace correspondiente. (*Imagen 4*)



Imagen 4: imagen que nos redirigirá a la página de descarga.

1. Una vez redirigidos a la página oficial de descarga clicamos sobre el enlace recomendado *(Imagen 5).* Y descargamos el archivo en el directorio de nuestra elección. *(Imagen 6)* En nuestro caso será en el Escritorio.

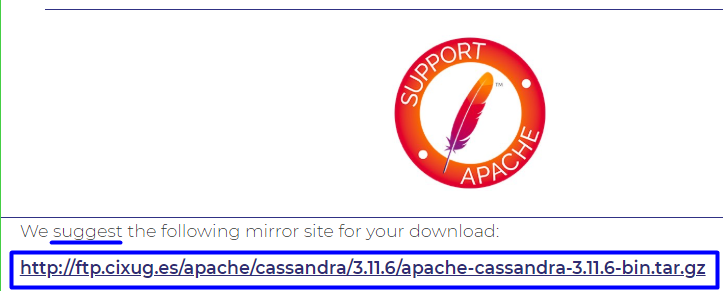


Imagen 5: enlace de descarga que elegiremos.

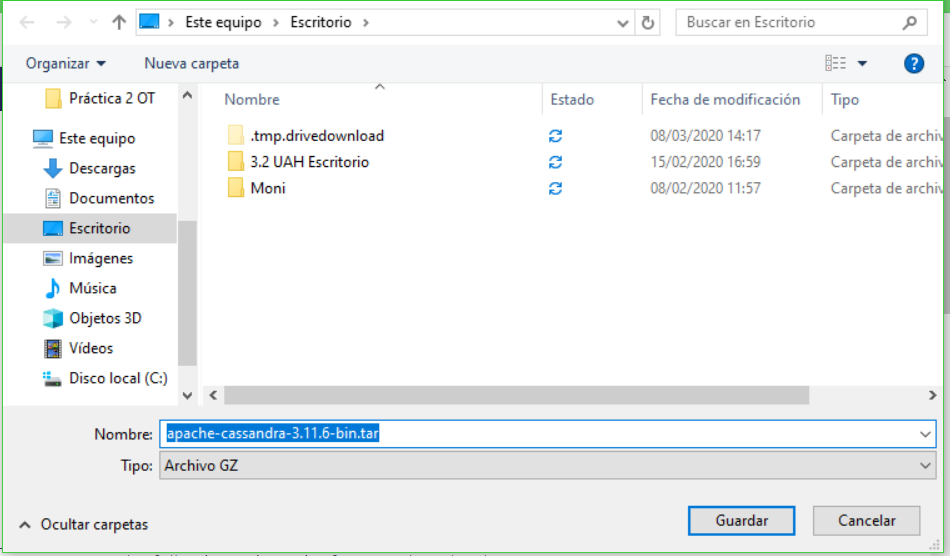


Imagen 6: descargamos el archivo en el directorio que queramos.

1. Como el archivo tiene una extensión “.gz”, necesitaremos descomprimirlo dos veces *(Imagen 7):* (para descomprimir el archivo hemos usado el programa 7-Zip)

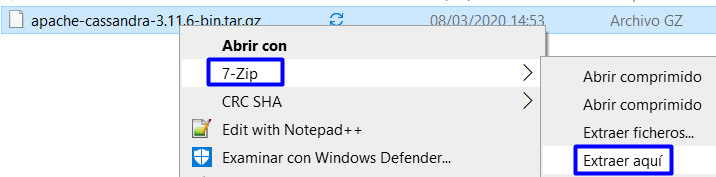


Imagen 7: cómo extraer el archivo.

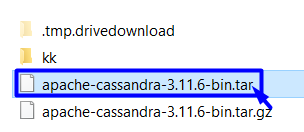
* 1. Extraemos el archivo con la extensión “.tar”. *(Imagen 8)*

Imagen 8: archivo generado tras la primera descompresión.

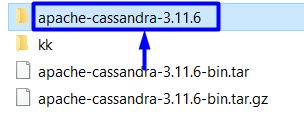
* 1. Extraemos la carpeta “apache-cassandra-3.11.6-bin”. *(Imagen 9)* Pero esta vez en extraemos la carpeta en el directorio “C:\”

Imagen 9: carpeta final. Extraída tras la segunda descompresión.

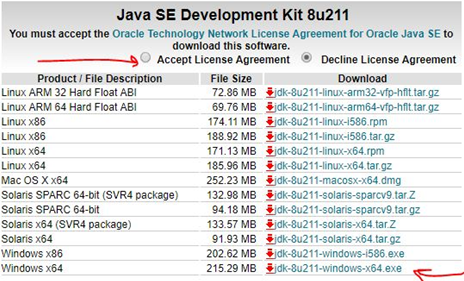
1. **Descargamos la última versión del JDK de Java 8 en el enlace:<https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>, aceptamos los términos y seleccionamos el archivo que más se adapte a nuestras necesidades *(Imagen 10).*

Imagen 10: página de descarga de Java.

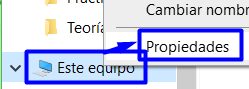
1. Una vez descargado el archivo, lo ejecutamos e instalamos.
2. El siguiente paso es añadir la variable JAVA\_HOME a las variables del entorno de Windows. Para ello, debemos seguir los siguientes pasos:
   1. ** Abrir el explorador de archivos, en el menú lateral izquierdo, hacer clic derecho sobre “Este equipo” y en el menú que se despliega elegir “Propiedades”. *(Imagen 11)*

Imagen 11: opción que nos redirecciona a las propiedades de nuestro equipo de Windows.

1. En la ventana que nos aparece, hacemos clic en la opción del menú lateral izquierdo “Configuración avanzada del sistema”. (Imagen 12)

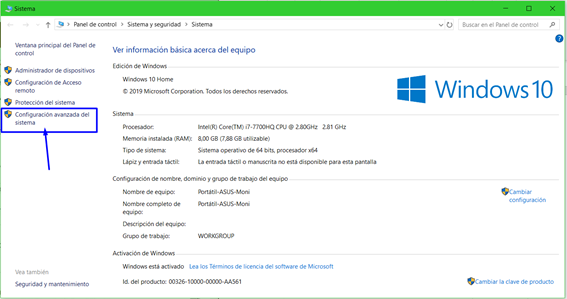


Imagen 12: opción que nos redirecciona a la ventana para editar las variables de entorno de Windows 10.

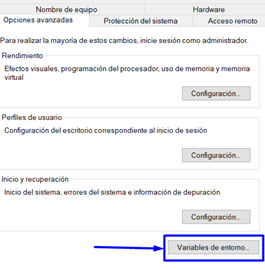
1. En la ventana que aparece, clicamos en “Variables de entorno…” (Imagen 13)

Imagen 13: opción para ver las variables del entorno.

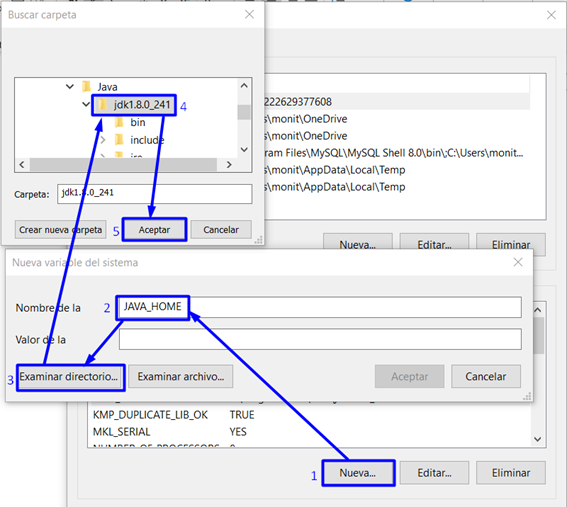
1. Creamos la variable “JAVA\_HOME” siguiendo los pasos que se observan en la siguiente imagen: (Imagen 14). De esta forma, podemos observar que la variable se ha añadido a las variables del sistema. (Imagen 15)

Imagen 14: pasos a seguir para crear la variable JAVA\_HOME.

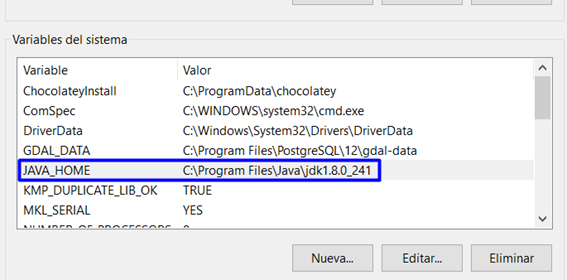


Imagen 15: variable JAVA\_HOME una vez añadida.

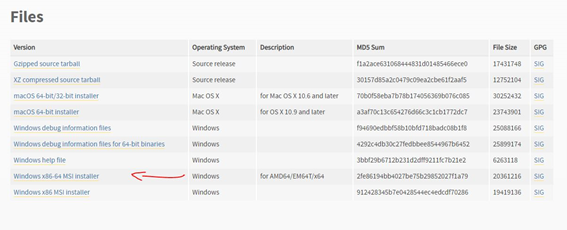
1. A continuación, necesitamos descargar Python 2.7 (esta versión es la compatible con CassandraDB a día de hoy). Podremos descargar el ejecutable de Python adecuado para nuestro sistema operativo desde el enlace (Imagen 16): <https://www.python.org/downloads/release/python-2716/>

Imagen 16: nosotros elegiremos la versión para Windows 10 de 64 bits.

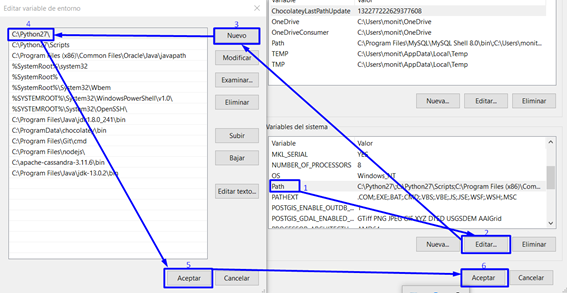
1. Una vez descargado el archivo .exe, lo abrimos, ejecutamos y seguimos los pasos de instalación.
2. ** Después de esto, debemos volver al menú de las variables del entorno *(Imagen 11)*. Pero esta vez, editaremos la variable del sistema “Path”. Para realizar la edición necesaria, debemos seguir los pasos que se muestran en la imagen *(Imagen 17).*

Imagen 17: pasos a seguir para Editar la variable del sistema Path para añadir Python 2.7

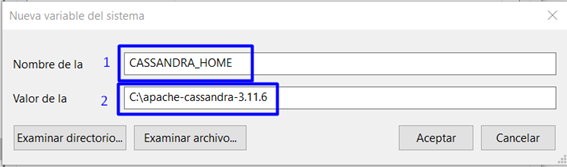
1. Ahora, vamos a hacer lo mismo que en los pasos 11 con Java y en el paso 14 con Python, pero con CassandraDB, es decir, vamos a añadir la variable CASSANDRA\_HOME *(Imagen 18)* y editar la variable Path *(Imagen 19)*.

Imagen 18: añadimos la variable CASSANDRA\_HOME.

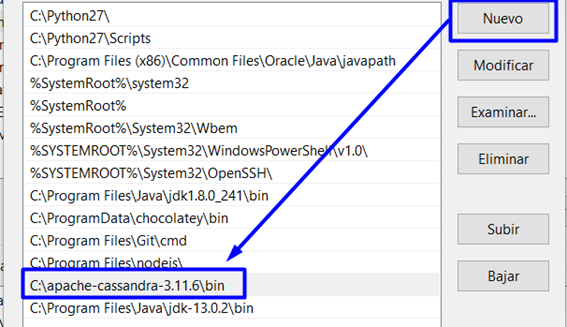


Imagen 19: editamos la variable Path y añadimos la ruta de instalación de CassandraDB.

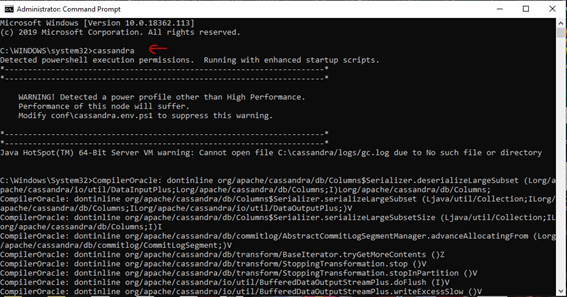
1. **Cerramos todas las ventanas de las variables del entorno y abrimos un terminal en Windows como administrador (esto se hace tecleando “cmd” en el buscador de Windows, darle clic derecho y elegir “Abrir como administrador”) y a continuación tecleamos: “cassandra” y nos aparecerá lo siguiente *(Imagen 20):*

Imagen 20: pantalla que aparece al ejecutar el comando cassandra.

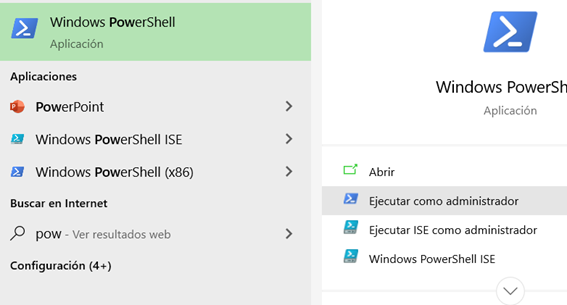
1. Si al ejecutar el comando salta algún error, debemos abrir la PowerShell de Windows como administrador *(Imagen 21)* y ejecutar el comando “Set-ExecutionPolicy Unrestricted” *(Imagen 22*) a continuación escribimos “O”.

Imagen 21: abrimos la PowerShell

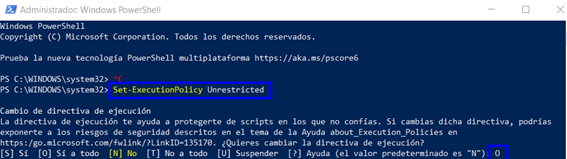


Imagen 22: pasos a seguir en la PowerShell

1. Una vez realizados estos pasos, debe aparecernos la pantalla de la *Imagen 18* y ya tendremos CassandraDB instalado y correctamente configurado en nuestro ordenador.

# PUESTA EN MARCHA

En este punto, nos centraremos en la preparación de un entorno de trabajo para trabajar con CassandraDB. Que se compondrá del despliegue de la base de datos a través de un Docker y el uso de Notebooks para realizar las conexiones y ejecutar las consultas.

El contenedor de Docker que utilizaremos es uno propio de la empresa Data Stack (la que mantiene hoy en día CassandraDB).

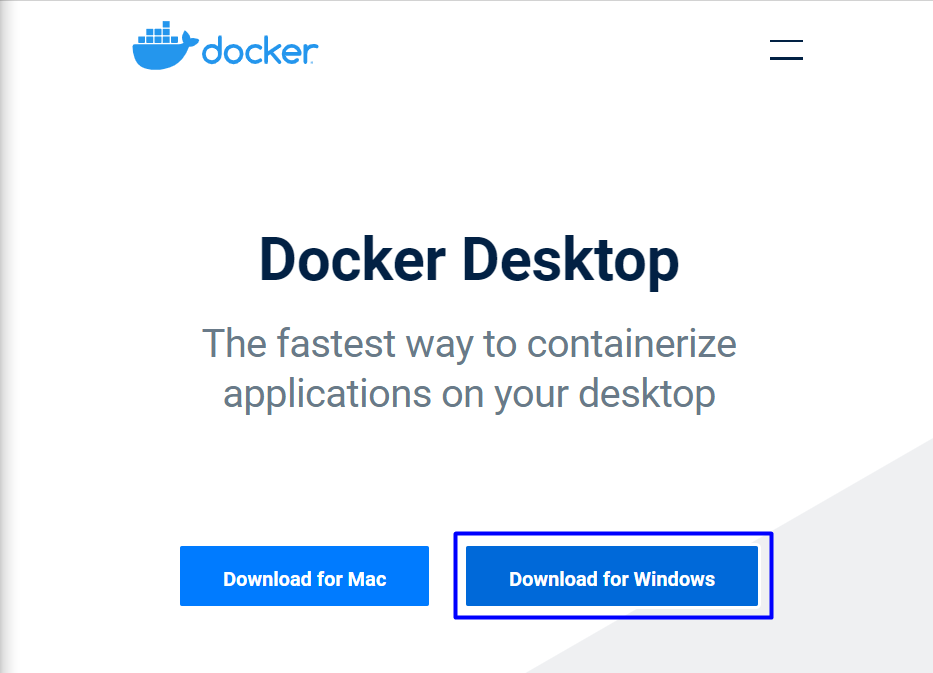
El primer paso es acceder a la página web de Docker ([https://www.docker.com/products/docker-deskto](https://www.docker.com/products/docker-desktop)), registrarnos si no tenemos una cuenta y descargarnos la herramienta seleccionando nuestro sistema operativo *(Imagen 23) (Imagen 24)*.

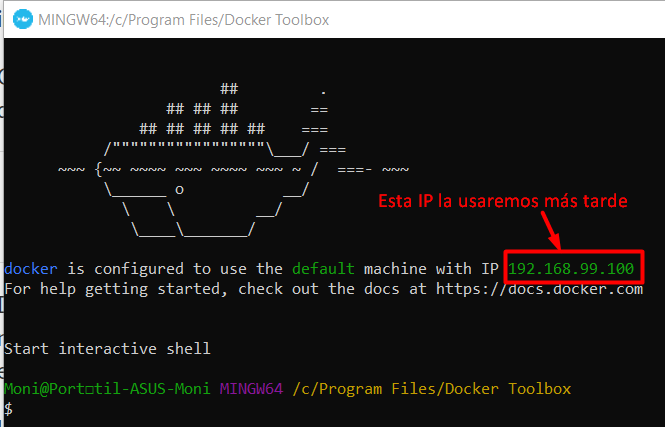
Imagen 23: botón de descarga de Docker para Windows.



Imagen 24: botón que descarga el ejecutable en nuestro ordenador.

Después, abrimos el ejecutable que nos hemos descargado y seguimos los pasos para la instalación.

Una vez instalado, necesitamos abrir el terminal de Docker (Imagen 25)

Imagen 25: el terminal una vez abierto y cargado.

Una vez abierto, lanzamos el comando para loguearnos:

docker login

Y seguimos los pasos que nos indica el terminal. (Imagen)

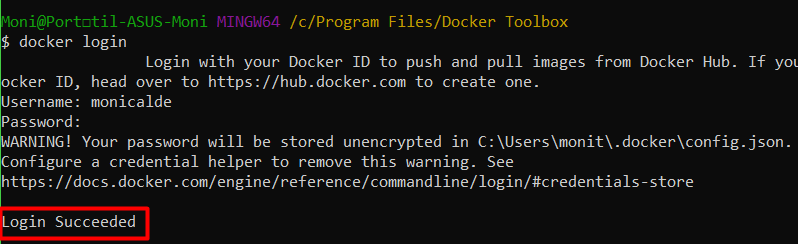


Imagen 26: proceso a seguir para hacer "login" desde el terminal.

Después de loguearnos, ejecutamos los siguientes comandos:

docker pull datastax/dse-server:6.7.7

docker pull datastax/dse-studio:latest

Estos comandos nos permiten hacer el pull del servidor Cassandra y el del notebook.

Y después de esto, lanzamos el contenedor con el siguiente comando:

docker run -p 9042:9042 -e DS\_LICENSE=accept --memory 4g --name dockerCassandra -d datastax/dse-server:6.7.7 -g -s -k

Con este comando, lo que hacemos es:

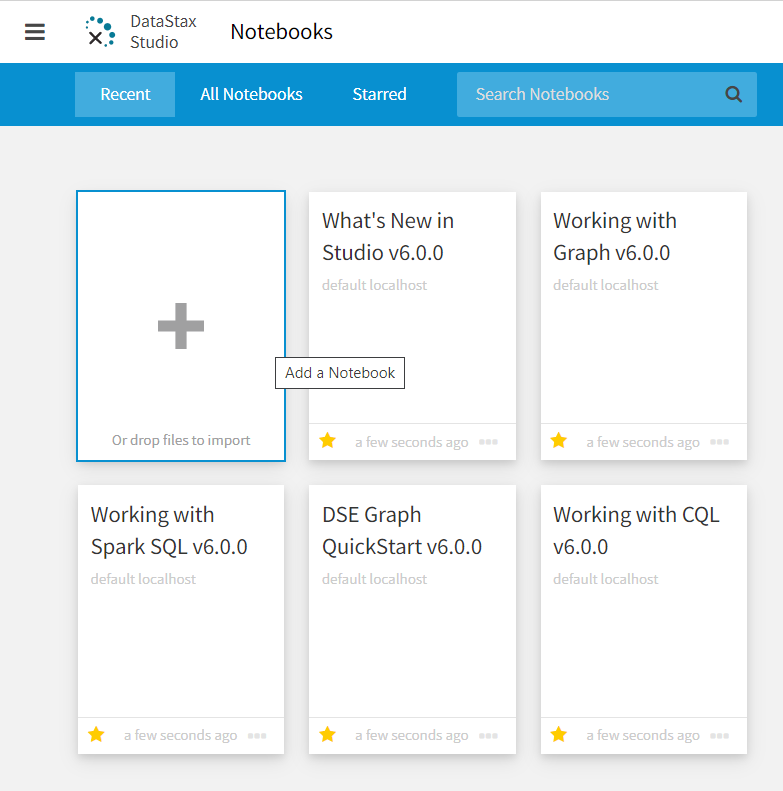
1. Exponer el puerto 9042, que es el puerto donde se está ejecutando Cassandra.
2. Le damos un nombre al contenedor (en nuestro caso: nombre)
3. Indicamos la imagen de la que acabamos hacer el pull (en nuestro caso: datastax/dse-server)

A continuación, lanzamos el segundo contenedor con este comando:

docker run -e DS\_LICENSE=accept --memory 4g --link dockerCassandra -p 9091:9091 --memory 1g --name docker2Cassandra -d datastax/dse-studio

Esto nos permite habilitar el puerto 9091 donde lo vamos a ejecutar, así como darle un nombre (nombre2) e indicar la imagen del contenedor (datastax/dse-estudio).

Una vez hecho esto, ya está todo listo y corriendo. Y para comprobarlo solo tenemos que acceder a la url: <http://192.168.99.100:9091>*(Imagen 27)* (La dirección IP de la URL es la que aparecía en la *Imagen 25*) donde veremos que tenemos el notebook listo para ejecutar.

Imagen 27: página que aparece al acceder a la URL de Docker

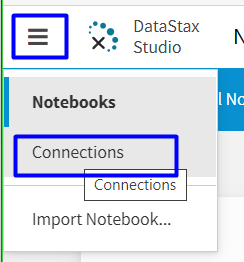
Si ya tenemos todo listo, pasamos a conectar nuestra base de datos de Cassandra. Para ello, seleccionamos la opción “Connections” del menú lateral, *(Imagen 28)*

Imagen 28: acceso a las conexiones de Docker.

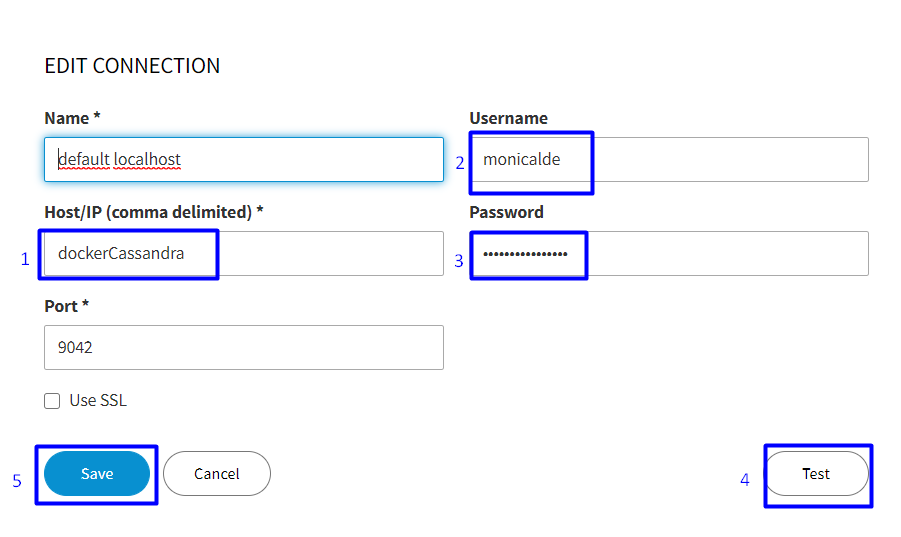
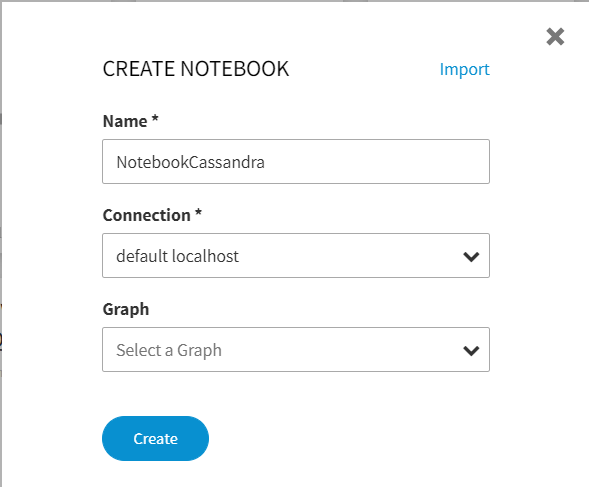
Una vez dentro de Connections, elegimos la que nos aparece por defecto, cambiamos los campos de Host/IP por el nombre que le habíamos asignado (dockerCassandra) y escribimos nuestro nombre de usuario y contraseña. *(Imagen 29)*

Imagen 29: pasos a seguir para establecer la conexión con Cassandra

Clicando la opción Test, nos debe aparecer un mensaje que diga: “Connected succesfully”. Una vez configurado esto, hacemos clic en Save.

Ahora ya podemos crear un Notebook donde podremos ejecutar las consultas. Para ello, en el menú lateral seleccionamos “Notebooks” y después al “+”.

Una vez nos aparecen las opciones para crear un nuevo Notebook, le ponemos el nombre que queramos, seleccionamos la conexión por defecto y hacemos clic en “Create”. *(Imagen 30)*

Imagen 30: opciones para crear un Notebook.

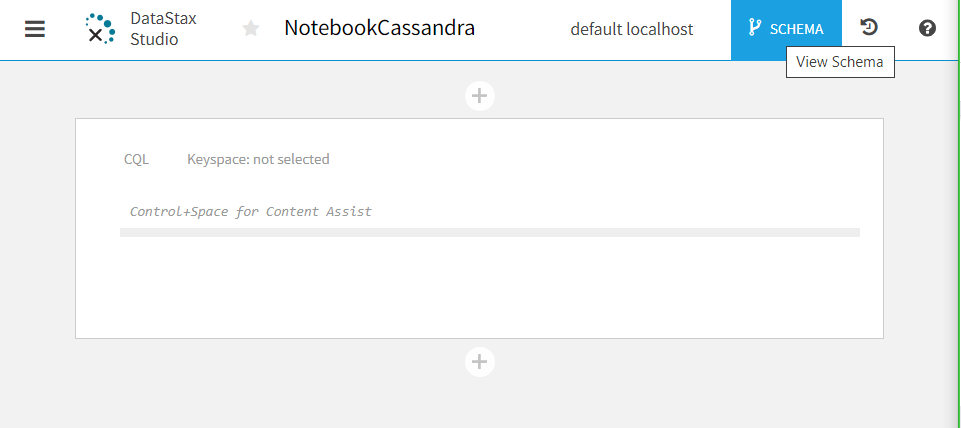
Una vez hecho todo esto, nos aparece el Notebook *(Imagen 31)* donde podremos empezar a crear las consultas CQL de nuestra base de datos Cassandra.

Imagen 31: Notebook recién creado en Docker.

Cabe destacar, que los Notebooks para esta conexión, ofrecen una ayuda para los comandos de las consultas pulsando las teclas “ctrl+espacio *(Imagen 32)* lo cual es muy útil al tratarse de una conexión con una base de datos de Cassandra que utiliza un sistema de consultas CQL (sistema explicado anteriormente).

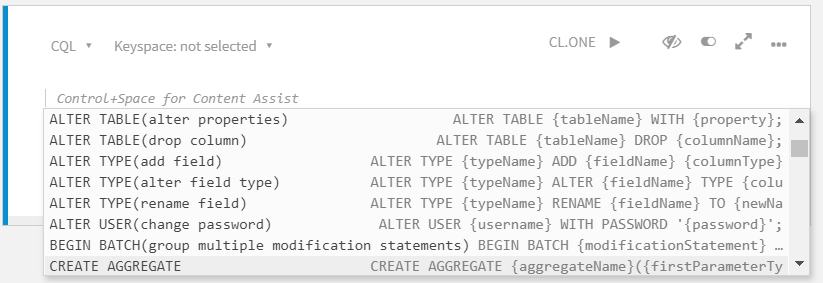


Imagen 32: ayuda de Docker para crear consultas (en este caso, de la forma CQL).

Una vez que hemos realizado las operaciones necesarias en nuestro entorno, para detener los contenedores desde el terminal de Docker, solo necesitamos ejecutar los siguientes comandos:

docker stop docker2Cassandra

docker stop dockerCassandra

Y para volver a iniciarlos, no es igual que antes, si no que debemos usar estos otros comandos:

docker run dockerCassandra

docker run docker2Cassandra

Para comprobar si el contenedor está levantado, podremos hacerlo con:

docker ps

# ENUMERACIÓN Y EXPLICACIÓN DE CASOS DE USO PARA LOS QUE ESTÁ PENSADA

Se debe utilizar Cassandra en el caso en que se necesite obligatoriamente una base de datos distribuida. Si solo vas a necesitar un nodo, hay alternativas mucho mejores. La principal ventaja de Cassandra es su capacidad de replicación a través de los nodos del sistema. También es apta para el uso en datacenters, pues su arquitectura está pensada para almacenar los datos en los nodos más cercanos a donde se haya hecho la petición de escritura en la base de datos.

Cassandra se utiliza mucho para el ecommerce, la detección de fraude y el análisis en tiempo real. Es su elección si vamos a almacenar datos semi estructurados y esperamos un crecimiento exponencial de la base de datos.

# ENUMERACIÓN DE LOS CASOS DE ÉXITO

## Digg

La empresa Digg es un sitio web sobre noticias de ciencia y tecnología. En 2009 decidieron migrar de MySQL a Cassandra. En palabras de Ian Eure, un ingeniero que trabajó en Digg en 2008-2010:

*“Después de considerar HBase, Hypertable, Cassandra, Tokyo Cabinet/Tyrant, Voldemort y Dynomite, nos quedamos con Cassandra. Cada sistema tiene sus puntos fuertes y sus debilidades, pero Cassandra es una buena mezcla de todo. Ofrece almacenamiento orientado a columnas, por lo que tienes algo más de estructura que los acercamientos de clave/valor. Opera en un clúster distribuido, de alto rendimiento y peer-to-peer. Y aunque le faltan algunas características necesarias, nos deja más cerca de donde queremos llegar que otras soluciones.*

*[…] El problema fundamental es algo endémico a la mentalidad de bases de datos relacionales, que establecen un mayor peso de computación en las lecturas en lugar de en las escrituras. Esto es algo totalmente equívoco cuando estamos hablando de aplicaciones web a gran escala, donde el tiempo de respuesta es crítico. Cada componente de la página bloquea las lecturas del almacén de datos. […] Las bases de datos no relacionales le dan la vuelta a este modelo, ya que no ejecutan operaciones complejas de lectura mediante SQL. El modelo te fuerza a cambiar el esfuerzo de computación a las escrituras, mientras que las lecturas se reducen a las operaciones más simples posibles.”*

## Twitter

En palabras de Ryan King, ingeniero de Twitter, esta es la explicación de la migración a Cassandra:

*“Tenemos una gran cantidad de datos, y con un factor de crecimiento muy elevado y encima acelerándose. Tenemos un sistema con MySQL+ memcache, pero se está convirtiendo en algo costosamente prohibitivo en términos de esfuerzo (personal trabajando para el sistema). Necesitamos un sistema que pueda crecer de una forma más automatizada y que presente alta disponibilidad.*

*[…] Las principales razones por las que migramos a Cassandra se resumen en: 1) No tiene puntos de fallo, 2) las escrituras son altamente escalables y 3) una comunidad open source saludable y productiva.”*

# CONCLUSIÓN

Las bases de datos no estructuradas ofrecen una solución a un problema creciente de almacenamiento de petabytes de datos que se generan diariamente. En la actualidad, existen decenas de opciones por las que decantarse. En este caso, Cassandra es una de las mejores opciones en su terreno, por no decir la mejor. Si nos encontramos ante un problema de escalado masivo de datos y de extrema disponibilidad, Cassandra es nuestra mejor opción. Si, por el contrario, queremos disponer de un sistema más pequeño y controlado, existen opciones más interesantes en el mercado.

# BIBLIOGRAFÍA

Sitio Web oficial de Apache Cassandra

<https://cassandra.apache.org/>

Blog sobre Datastax (desarrollador actual de Cassandra)

<https://www.datastax.com/blog/2013/01/2012-review-performance>

Proyecto de fin de ciclo de Juan José López Roldán sobre Cassandra y NoSQL:

<http://informatica.gonzalonazareno.org/proyectos/2017-18/Memoria-Proyecto-juanjoselopezroldan.pdf>

Modelo de datos y estructura de CassandraDB:

<https://sites.google.com/site/uegoman/modelo-de-datos-de-cassandra>

Resumen sobre varios aspectos de Cassandra

<https://www.panel.es/blog/base-de-datos-cassandra/>

Artículo Web sobre Cassandra:

<https://josek.net/2010/03/cassandra-nsql/>

Definición de CQL:

<https://es.wikipedia.org/wiki/Contextual_Query_Language>

Consultas CQL (Ejemplos):

<http://abiasforaction.net/a-practical-introduction-to-cassandra-query-language/>

Artículo sobre como instalar CassandraDB en Windows 10 (en inglés):

<https://medium.com/@ciceroaferreira/how-to-install-cassandra-db-177e27226cdc>

Página de instalación de CassandraDB para una versión más reciente:

<https://www.how2shout.com/how-to/install-apache-cassandra-on-windows-10-8-7-without-datastax.html>

Documento que nos explica lo que es una base de datos NoSQL:

<https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>

Página dónde podemos encontrar la definición de base de datos distribuida:

<https://es.wikipedia.org/wiki/Base_de_datos_distribuida>

Tutorial para preparar el entorno de trabajo de Cassandra con un Docker:

<https://openwebinars.net/blog/preparar-entorno-trabajo-cassandra/?utm_source=youtube&utm_medium=descripcion&utm_campaign=preparar-entorno-trabajo-cassandra>