Informe de Laboratorio Nro 07: Contenedor de Microsoft SQL Server

Estrella Palacios, Katherine Lizbeth (2016056193)

Universidad Privada de Tacna

Escuela Profesional de Ingeniería de Sistemas

Tacna, Perú

---

1. INFORMACIÓN GENERAL

Objetivos:

- Crear un contenedor de la imagen de Microsoft SQL Server
- Desplegar una BD usando un contenedor

Equipos y programas utilizados: Para el siguiente laboratorio requerimos de:

- PC con sistema operativo Windows 10
- Microsoft SQL Server Management Studio
- Docker Desktop

2. MARCO TEÓRICO

2.1. ***Historia***

Salomon Hykes comenzó Docker comenzó como un proyecto interno dentro de dotCloud,empresa enfocada a PaaS (plataforma como servicio). Fué liberado como código abierto en marzode 2013.Con el lanzamiento de la versión 0.9 (en marzo de 2014) Docker dejó de utilizar LXC comoentorno de ejecución por defecto y lo reemplazó con su propia librería, libcontainer (escrita en Go),que se encarga de hablar directamente con el kernel.Actualmente es uno de los proyectos con más estrellas en GitHub, con miles debifurcaciones (forks) y miles de colaboradores.

2.2. ***Características***

Las principales características de Docker son:

- **Portabilidad:** el contenedor Docker podemos desplegarlo en cualquier sistema, sin necesidad de volver a configurarlo o realizar las instalaciones necesarias para que la aplicación funcione, ya que todas las dependencias son empaquetadas con la aplicación en el contenedor.
- **Ligereza:** los contenedores Docker sólo contienen lo que las diferencia del sistema operativo en el que se ejecutan, no se virtualiza un SO completo.
- **Autosuficiencia:** un contenedor Docker no contiene todo un sistema operativo completo,sólo aquellas librerías, archivos y configuraciones necesarias para desplegar las funcionalidades que contenga.

2.3. ***Ventajas y Desventajas***

Usar contenedores Docker permite a des arrolladores y administradores de sistemas probar aplicaciones o servicios en un entorno seguro e igual al de producción, reduciendo los tiempos de pruebas y adaptaciones entre los entornos de prueba y producción. Las principales ventajas de usar contenedores Docker son:

- Las instancias se inician en pocos segundos.
- Son fácilmente replicables.
- Es fácil de automatizar e integrar en entornos de integración continua.
- Consumen menos recursos que las máquinas virtuales tradicionales.
- Mayor rendimiento que la virtualización tradicional ya que corre directamente sobre el Kernel de la máquina en la que se aloja, evitando al hypervisor.
- Ocupan mucho menos espacio.
- Aisla las dependencias de una aplicación de las instaladas en el host.
- Existe un gran repositorio de imágenes ya creadas sobre miles de aplicaciones, que además pueden modificarse libremente.

Por todo esto Docker ha entrado con mucha fuerza en el mundo del desarrollo, ya que permite desplegar las aplicaciones en el mismo entorno que tienen en producción o viceversa,permite desarrollarlas en el mismo entorno que tendrán en producción. Aunque también tiene algunas desventajas:

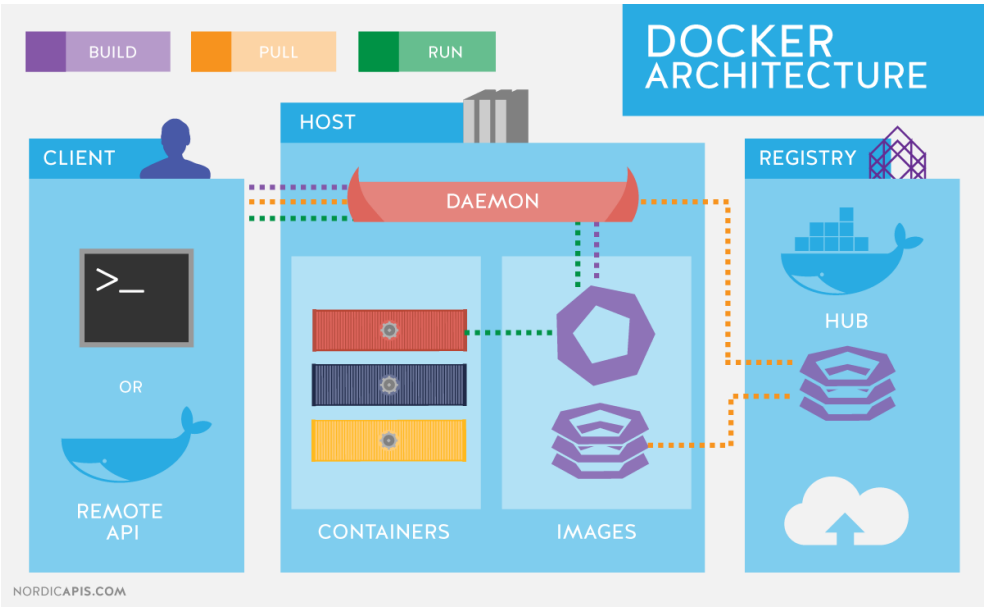
- Sólo puede usarse de forma nativa en entornos Unix con Kernel igual o superior a 3.8.
- Sólo soporta arquitecturas de 64 bits.
- Como es relativamente nuevo, puede haber errores de código entre versiones.

2.4. ***Arquitectura***

Docker usa una arquitectura cliente-servidor. El cliente de Docker habla con el demonio de Docker que hace el trabajo de crear, correr y distribuir los contenedores. Ambos pueden ejecutarse en el mismo sistema, o se puede conectar un cliente a un demonio Docker remoto. El cliente Dockery el demonio se comunican vía sockets o a través de una RESTfull API. (imagen pagina oficial.

Explicamos un poco por orden la arquitectura o funcionamiento de Docker:

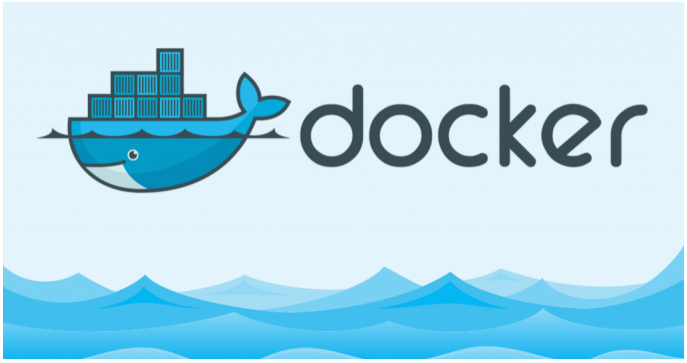
- El cliente de Docker (Docker Client) es la principal interfaz de usuario para Docker. Él acepta comandos del usuario y se comunica con el demonio Docker.
- El demonio Docker (Docker Engine) corre en una máquina anfitriona (host). El usuario no interactúa directamente con el demonio, en su lugar lo hace a través del cliente Docker.
- El demonio Docker levanta los contenedores haciendo uso de las imágenes, que puedenestar en local o en el Docker Registry.
- Cada contenedor se crea a partir de una imagen y es un entorno aislado y seguro dónde seejecuta nuestra aplicación.



*2.5. **Componentes***

Según la documentación oficial, Docker tiene dos principales componentes:

- **Docker:** Plataforma open source de virtualización con contenedores.
- **Docker Hub:** Plataforma de Software como servicio (SaaS, Software-as-a-Service) para compartir y administrar contenedores Docker.



Pero también necesitamos conocer otros componentes y conceptos:

- **Docker Engine:** Es el demonio que se ejecuta dentro del sistema operativo (Linux) y que expone una API para la gestión de imágenes, contenedores, volúmenes o redes.
- **Docker Client:** Cualquier software o herramienta que hace uso de la API del demonio Docker, pero suele ser el comando docker, que es la herramienta de línea de comandos para gestionar Docker Engine.Éste cliente puede configurarse para hablar con un Docker local o remoto, lo que permite administrar nuestro entorno de desarrollo local como nuestros servidores de producción.
- **Docker Images:** Son plantillas de sólo lectura que contienen el sistema operativo base (más adelante entraremos en profundidad) dónde correrá nuestra aplicación, además de las dependencias y software adicional instalado, necesario para que la aplicación funcione correctamente. Las plantillas son usadas por Docker Engine para crear los contenedores Docker.
- **Docker Registries:** Los registros de Docker guardan las imágenes. Pueden ser repositorios públicos o privados.El registro público lo provee el Hub de Docker, que sirve tanto imágenes oficiales cómo las subidas por usuarios con sus propias aplicaciones y configuraciones.
- **Docker Containers:** El contenedor de Docker aloja todo lo necesario para ejecutar un servicio o aplicación. Cada contenedor es creado de una imagen base y es una plataforma aislada.
- **Docker Compose:** Es otro proyecto open source que permite definir aplicaciones multi-contenedor de una manera sencilla. Es una alternativa más cómoda al uso del comando docker run, para trabajar con aplicaciones con varios componentes.
- **Docker Machine:** Es un proyecto open source para automatizar la creación de máquinas virtuales con Docker instalado, en entornos Mac, Windows o Linux, pudiendo administrar así un gran número de máquinas Docker.Incluye drivers para Virtualbox, que es la opción aconsejada para instalaciones de Docker en local, en vez de instalar Docker directamente en el host.

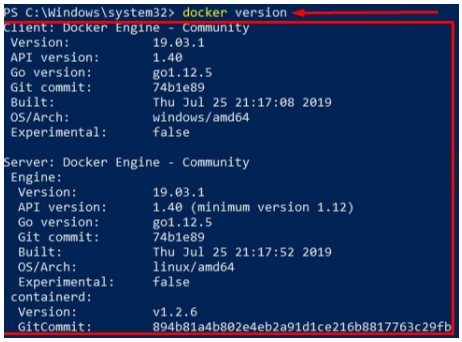
3. PROCEDIMIENTO

Paso 1 : Iniciar Docker

a) Iniciaremos con nuesytra cuenta docker en Docker Deskot.

b) Debemos estar registrados para ingresar.

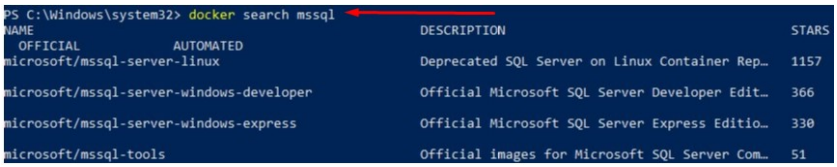
c) Ejecutamos PowerShell y agregamos el siguiente comando **docker version**.

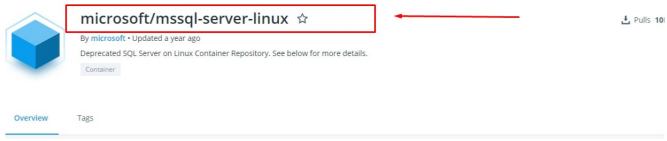


Paso 2 : Crearemos un contenedor con Microsoft SQL Server para Linux

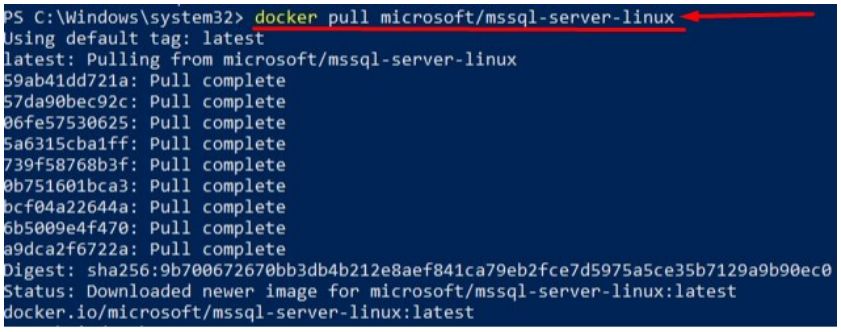
d) Ejecutaremos el siguiente comando **docker search mssql** en PowerShell.

e) Ingraseremos nuestra cuenta en la página web Desktop Hub y buscaremos el repositorio **"microsoft/mssql-server-linux"**.





f) Copiando el comando en la aplicación PowerShell (docker pull microsoft/mssql-server-linux) Se descargará la imagen del contenedor de Microsoft SQL Server en un servidor Linux como podemos ver en la imagen.



g) Verificaremos la imagen descargada con el siguiente comando **docker images**.

h) Iniciaremos el contenedor con el siguiente comando

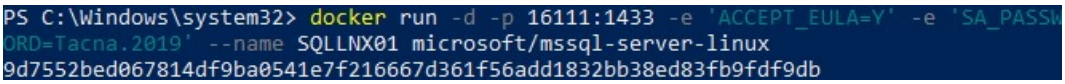
i) Nos devolcera el ID del contenedor.

9d7552bed067814df9ba0541e7f216667d361f56add1832bb38ed83fb9fdf9db

j) Verificamos que el contenedor se está ejecutando correctamente con el siguiente comando docker ps. El resultado será similar al siguiente:

k) Abriremos el programa Microsoft SQL Server Management Studio 17 y conectamos con los siguientes datos:

Nombre del servidor : 127.0.0.1,16111



Inicio de sesión :sa

Contraseña:Epis2019

l) Iniciaremos una consulta ingresando lo siguiente :

SELECT @@VERSION

m) En la aplicación PowerShell ejecutamos el siguiente comando:

docker rm -f SQLLNX01

n) Verificamos la eliminación del contenedor con el siguiente comando:

docker ps

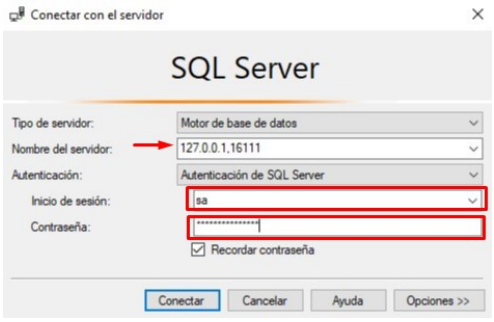
4. ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

a) Con el comando para iniciar con un contenedor podemos asignar los siguientes parámetros:

-p : Asignar el puerto.

'sapassword' : Contraseña del Inicio de Sesion SQL, usuario sa.

–name : Nombre del contenedor.



5. CUESTIONARIO

a) **¿Con qué comando(s) exportaría la imagen de Docker de Microsoft SQL Server a otra PC o servidor?**

Exportar la Imagen de Docker de Microsoft SQL Server "docker export (ID contenedor) ¿Nombreimagen.tar"

docker
9d7552bed067814df9ba0541e7f216667d361f56add1832bb38ed83fb9fdf9db
¿SQL.tar

```
docker export 9d7552bed067814df9ba0541e7f216667d361f56add1832bb38ed83fb9fdf9db > SQL.tar
```

b) **¿Con qué comando(s) podría generar dos volúmenes para un contenedor para distribuir en un volumen el Archivo de Datos (.mdf) y en otro el Archivo Log (.ldf)?**

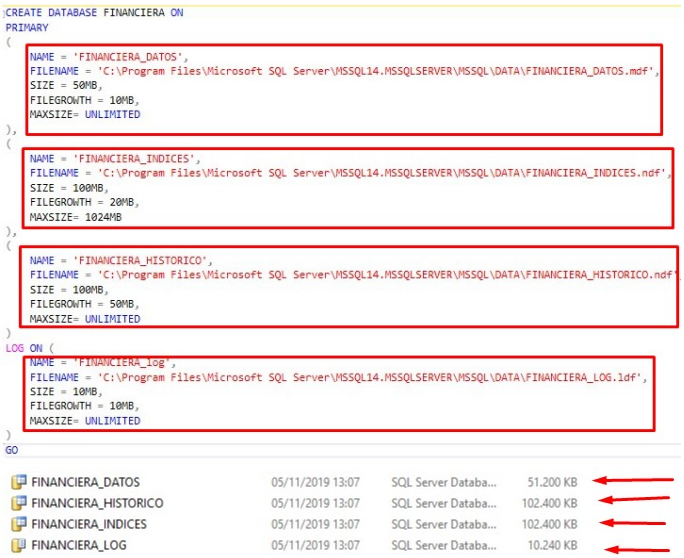
CREATE DATABASE NAMEDATABASE ON
(FILENAME = N'/var/opt/mssql/data2/NDATABASE.mdf'),
(FILENAME = N'/var/opt/mssql/data2/NDATABASElog.ldf')
FOR ATTACH
GO

c) **Genere un nuevo contenedor y cree la base de datos con las siguientes características.**

Nombre : FINANCIERA
Archivos:

- DATOS (mdf) : Tamaño Inicial : 50MB, Incremento: 10MB, Ilimitado
- INDICES (ndf) Tamaño Inicial : 100MB, Incremento: 20MB, Maximo: 1GB
- HISTORICO (ndf) Tamaño Inicial : 100MB, Incremento: 50MB, Ilimitado
- LOG (ldf) Tamaño Inicial : 10MB, Incremento: 10MB, Ilimitado

¿Cuál sería el script SQL que generaría esta base de datos?



6. CONCLUSIONES

Los contenedores de Docker nacen a partir de una imagen y en estos contenedores podemos solo ejecutar e instalar servicios, viene siendo como crear una maquina virtual a partir de una imagen (snapshot) pero muchísimo más ligera.