

Taller Arquitectura

Michael Stiven Castillo Castillo Id 825947

Corporación Universitaria Minuto de Dios

Ingeniería de Sistemas,

Arquitectura De Sofrware

Alexander Matallana Porras

21 de marzo del 2025

1. Diferencia entre archivos YAML y JSON

Ambos formatos, YAML y JSON, son maneras de estructurar y representar datos en forma de texto. Sin embargo, presentan diferencias clave en su sintaxis y en el contexto en el que se utilizan. JSON se caracteriza por su uso de caracteres específicos como llaves, corchetes y comas para definir objetos y listas, lo que lo hace muy popular para la transmisión de datos entre aplicaciones, especialmente en el ámbito de las APIs y servicios web. En contraste, YAML se basa en la indentación y la estructura jerárquica sin depender de caracteres de cierre, lo que lo convierte en una opción más intuitiva y legible para configurar aplicaciones y sistemas. Esta claridad en la estructura facilita la edición manual y reduce el riesgo de errores tipográficos, siendo ideal para archivos de configuración en entornos complejos.

2. Uso de docker-compose.yml

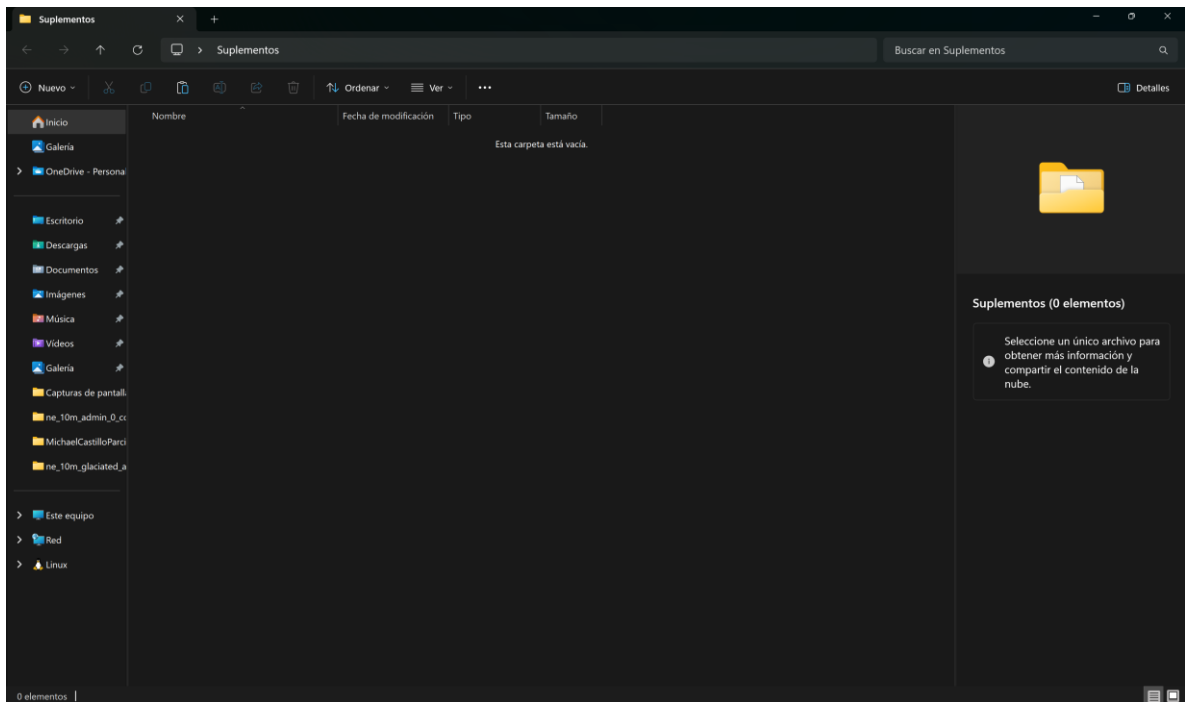
El archivo `docker-compose.yml` es fundamental cuando se trabaja con Docker Compose, una herramienta que permite orquestar y gestionar múltiples contenedores de forma simultánea. Este archivo centraliza la definición de cada contenedor, abarcando aspectos como la imagen a utilizar, las variables de entorno, la asignación de puertos, la configuración de volúmenes y la integración en redes específicas. La ventaja de esta aproximación radica en la simplificación del proceso de despliegue, ya que se puede levantar un conjunto completo de servicios con un único comando. Además, esta

configuración centralizada favorece la consistencia en los entornos de desarrollo, pruebas y producción, permitiendo replicar la infraestructura de manera confiable en diferentes sistemas.

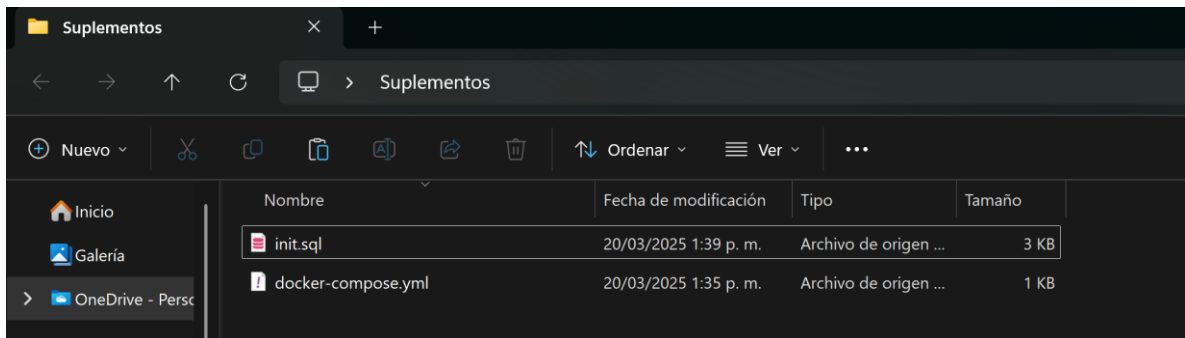
3. Creación de un contenedor usando un archivo YAML (docker-compose.yml)

El proceso para levantar un contenedor mediante Docker Compose se inicia asegurándose de que tanto Docker como Docker Compose estén correctamente instalados y configurados en el sistema. Una vez hecho esto, se procede a crear un archivo de configuración en formato YAML, en el que se describe en detalle cada servicio que se desea ejecutar, especificando las características necesarias para su correcto funcionamiento. Este archivo permite declarar de manera organizada las relaciones y dependencias entre contenedores, facilitando la administración de recursos y la comunicación entre ellos. La ejecución del comando correspondiente a Docker Compose desencadena la descarga de las imágenes necesarias (si aún no están disponibles localmente), la creación de los contenedores según las especificaciones y la configuración de redes y volúmenes, lo que se traduce en una infraestructura lista para operar de forma integrada y en paralelo.

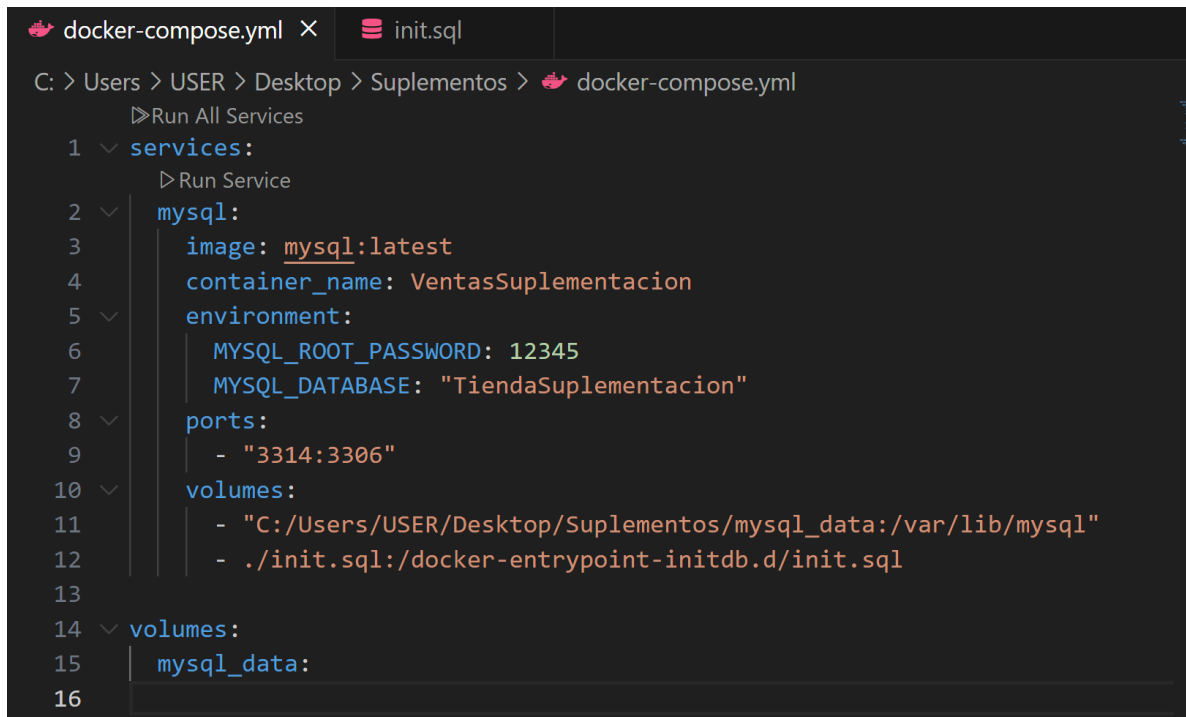
1. Se crea la carpeta Suplementos



2. Creamos dos documentos llamados init.sql y Docker-compose.yml



3. Agregamos la información tanto en Docker-compose.yml y en init.sql



```
docker-compose.yml X  init.sql
C: > Users > USER > Desktop > Suplementos > docker-compose.yml
  Run All Services
1  services:
  Run Service
2  mysql:
3    image: mysql:latest
4    container_name: VentasSuplementacion
5    environment:
6      MYSQL_ROOT_PASSWORD: 12345
7      MYSQL_DATABASE: "TiendaSuplementacion"
8    ports:
9      - "3314:3306"
10   volumes:
11     - "C:/Users/USER/Desktop/Suplementos/mysql_data:/var/lib/mysql"
12     - ./init.sql:/docker-entrypoint-initdb.d/init.sql
13
14  volumes:
15    mysql_data:
16
```

```

USE TiendaSuplementacion;

-- Tabla de clientes (no tiene dependencias, se puede crear primero)
CREATE TABLE IF NOT EXISTS clientes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    direccion VARCHAR(255),
    telefono VARCHAR(20)
);

-- Tabla de proveedores (no tiene dependencias, se puede crear también al
CREATE TABLE IF NOT EXISTS proveedores (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    telefono VARCHAR(20),
    email VARCHAR(100)
);

-- Tabla de productos (depende de proveedores)
CREATE TABLE IF NOT EXISTS productos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion VARCHAR(255),
    precio DECIMAL(10,2),
    proveedor_id INT,
    FOREIGN KEY (proveedor_id) REFERENCES proveedores(id)
);

-- Tabla de ventas (depende de clientes y productos)
CREATE TABLE IF NOT EXISTS ventas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    cliente_id INT,
    producto_id INT,
    fecha DATE,
    cantidad INT,
    FOREIGN KEY (cliente_id) REFERENCES clientes(id),
    FOREIGN KEY (producto_id) REFERENCES productos(id)
);

```

4. Insertamos los datos a las 4 tablas que ya hemos creado

```
-- Inserción de datos en proveedores
INSERT INTO proveedores (nombre, telefono, email) VALUES
('NutriFit', '555-1234', 'contacto@nutrifit.com'),
('SportMax', '555-5678', 'info@sportmax.com'),
('VidaSana', '555-9012', 'ventas@vidasana.com');

-- Inserción de datos en productos
INSERT INTO productos (nombre, descripcion, precio, proveedor_id) VALUES
('Proteína Whey', 'Proteína de suero de leche', 29.99, 1),
('Creatina Monohidratada', 'Creatina en polvo para rendimiento', 19.99, 2),
('Multivitamínico', 'Vitaminas y minerales esenciales', 14.99, 3);

-- Inserción de datos en clientes
INSERT INTO clientes (nombre, direccion, telefono) VALUES
('Carlos Pérez', 'Calle 123, Ciudad A', '555-0011'),
('Lucía Gómez', 'Avenida 456, Ciudad B', '555-0022'),
('Mario Rodríguez', 'Plaza 789, Ciudad C', '555-0033');

-- Inserción de datos en ventas
INSERT INTO ventas (cliente_id, producto_id, fecha, cantidad) VALUES
(1, 1, '2024-03-01', 2),
(2, 2, '2024-03-05', 1),
(3, 3, '2024-03-10', 3);
```

5. Usamos el comando docker-compose up -d para asociarlo con docker

```
C:\Users\USER\Desktop\Suplementos>docker-compose up -d
[+] Running 11/11
  ✓mysql Pulled                                25.2s
  ✓804bb8ae89de Pull complete                  8.6s
  ✓1b515e7ceb69 Pull complete                  8.6s
  ✓eaal1c0a9f08 Pull complete                  8.7s
  ✓8d18181893b8 Pull complete                  9.1s
  ✓e0a910cc8604 Pull complete                  9.2s
  ✓bc0c792ca096 Pull complete                  9.2s
  ✓8d73d2a73425 Pull complete                 10.6s
  ✓4a7e00d873b9 Pull complete                 10.6s
  ✓27a2553d6a80 Pull complete                 22.3s
  ✓69e76254f502 Pull complete                 22.3s
[+] Running 2/2
  ✓Network suplementos_default Created          0.1s
  ✓Container VentasSuplementacion Started      1.9s
C:\Users\USER\Desktop\Suplementos>
```

6. Ingresamos a Docker para verificar si el contenedor se creo

<input type="checkbox"/>		suplementos	-	-	-	0.64%	30 minutes ago			
<input type="checkbox"/>		VentasSuplementacion	4f34020a7297	mysql:latest	3314:3306	0.64%	30 minutes ago			

7. Podemos verificar que las tablas se pueden visualizar

TiendaSuplementacion					
Suplementos > mysql_data > TiendaSuplementacion					
<div> <div>Nuevo</div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div>Ordenar</div> <div>Ver</div> <div></div> </div>					
Inicio	Nombre	Fecha de modificación	Tipo	Tamaño	
Galería	clientes.ibd	21/03/2025 8:54 p. m.	Archivo IBD	112 KB	
OneDrive - Perso	productos.ibd	21/03/2025 8:54 p. m.	Archivo IBD	128 KB	
	proveedores.ibd	21/03/2025 8:54 p. m.	Archivo IBD	112 KB	
Escritorio	ventas.ibd	21/03/2025 8:54 p. m.	Archivo IBD	144 KB	

- Abrimos mysql y hacemos ingeniería inversa donde podremos ver la relación de cada tabla

