Laboratorio 9. Uso de DOCKER para la instalación de PostgreSQL + PGADMIN

¿Qué es un docker-compose?

`docker-compose.yaml` es un archivo de configuración utilizado con Docker Compose, una herramienta que permite definir y gestionar aplicaciones Docker multi-contenedor. Este archivo se utiliza para describir la configuración de los servicios, redes y volúmenes que componen una aplicación Docker, permitiendo a los usuarios definir la estructura de su aplicación de manera declarativa.
Parte A. El archivo docker-copose.yaml
A continuación, explicaremos el contenido de las líneas de código del archivo *docker-compose.yaml.*

¡Claro! Aquí tienes una explicación paso a paso del archivo `docker-compose.yaml`:

*1. **Versión de Compose:***
  ```*yaml*
  *version: "3.8"*
  ```
  Indica la versión de Docker Compose que se va a utilizar. En este caso, la versión es "3.8".

*2. **Servicios:***
  ```*yaml*
  *services:*
  ```
  Define los servicios que serán orquestados por Docker Compose. En este caso, hay dos servicios: `postgres` y `pgadmin`.

*3. **Servicio PostgreSQL (`postgres`):***
  ```*yaml*
   *postgres:*
     *image: postgres*
     *container_name: postgres*
     *restart: always*
     *ports:*
       *- "5432:5432"*
     *environment:*
       *POSTGRES_USER: admin*
       *POSTGRES_PASSWORD: admin*
       *POSTGRES_DB: mydb*
     *volumes:*
       *- pgdatadaw:/var/lib/postgresql/data*
       *- ./scriptdb.sql:/docker-entrypoint-initdb.d/scriptdb.sql*
  ```
  - `image`: Especifica la imagen Docker a utilizar, en este caso, "postgres".
  - `container_name`: Define el nombre del contenedor como "postgres".

- `restart: always`: Indica que el contenedor se reiniciará automáticamente en caso de fallo o reinicio del sistema.
- `ports`: Mapea el puerto local "5432" al puerto del contenedor "5432", permitiendo acceder a la base de datos PostgreSQL desde el host.
- `environment`: Configura las variables de entorno del contenedor, estableciendo un usuario, contraseña y nombre de la base de datos PostgreSQL.
- `volumes`: Mapea volúmenes para persistir datos, uno para la data de PostgreSQL y otro para un script SQL de inicialización.

*4. **Servicio pgAdmin (`pgadmin`):***
```yaml
pgadmin:
  image: dpage/pgadmin4
  container_name: pgadmin
  restart: always
  ports:
    - "8080:80"
  environment:
    PGADMIN_DEFAULT_EMAIL: admin@admin.com
    PGADMIN_DEFAULT_PASSWORD: admin
  volumes:
    - pgadmindaw:/var/lib/pgadmin
```
- `image`: Especifica la imagen Docker a utilizar, en este caso, "dpage/pgadmin4".
- `container_name`: Define el nombre del contenedor como "pgadmin".
- `restart: always`: Indica que el contenedor se reiniciará automáticamente en caso de fallo o reinicio del sistema.
- `ports`: Mapea el puerto local "8080" al puerto del contenedor "80", permitiendo acceder a la interfaz web de pgAdmin desde el host.
- `environment`: Configura las variables de entorno del contenedor, estableciendo un correo electrónico y contraseña para el usuario administrador de pgAdmin.
- `volumes`: Mapea un volumen para persistir datos de pgAdmin.

*5. **Volúmenes:***
```yaml
volumes:
  pgdatadaw:
  pgadmindaw:
```
Define los volúmenes que serán utilizados por los servicios `postgres` y `pgadmin` para persistir datos.

En resumen, este archivo `docker-compose.yaml` describe la configuración para orquestar dos servicios Docker: PostgreSQL y pgAdmin, junto con la definición de volúmenes para la persistencia de datos. Es especialmente útil para la configuración y despliegue consistente de ambientes de desarrollo y producción.

Revisar antes de ver con los alumnos.

Parte B.

Práctica Guiada con Docker y Docker Compose para PostgreSQL + pgAdmin

Objetivo:
- Instalar y configurar un entorno de desarrollo local para PostgreSQL y pgAdmin utilizando Docker y Docker Compose.

Paso 1: Preparación del Entorno
1. **Instalación de Docker y Docker Compose:**
   - Asegúrate de tener Docker y Docker Compose instalados en tu máquina. Si no los tienes, sigue las instrucciones para tu sistema operativo.

Paso 2: Descarga del Archivo `docker-compose.yaml`
1. **Descargar el Archivo `docker-compose.yaml`:**
   - Descarga el archivo `docker-compose.yaml` desde el enlace proporcionado o copia el contenido directamente.

Paso 3: Creación de Carpetas
1. **Crear una Carpeta para la Práctica:**
   - Crea una nueva carpeta en tu sistema para organizar los archivos relacionados con esta práctica.

Paso 4: Explicación del Archivo `docker-compose.yaml`
1. **Abrir Visual Studio Code:**
   - Abre Visual Studio Code y navega a la carpeta que creaste para la práctica.

2. **Crear un Nuevo Archivo `docker-compose.yaml`:**
   - Crea un nuevo archivo llamado `docker-compose.yaml` en Visual Studio Code.

3. **Copiar y Pegar el Contenido:**
   - Copia y pega el contenido del archivo `docker-compose.yaml` proporcionado en clase.

4. **Analizar el Contenido:**
   - Revisa cada sección del archivo y asegúrate de entender la configuración de servicios, variables de entorno y volúmenes.

Paso 5: Ejecución de Docker Compose
1. **Abrir la Terminal en Visual Studio Code:**
   - Abre la terminal integrada en Visual Studio Code.

2. **Navegar a la Carpeta de la Práctica:**
   - Utiliza el comando `cd` para cambiar al directorio de la práctica.

3. **Ejecutar Docker Compose:**
   - Ejecuta el siguiente comando para iniciar los servicios en segundo plano:
     ```bash
     docker-compose up -d
     ```

Paso 6: Verificación del Entorno
1. **Verificar Contenedores en Ejecución:**
   - Ejecuta el siguiente comando para asegurarte de que los contenedores estén en ejecución:
     ```bash
     docker ps
     ```

Paso 7: Acceso a PostgreSQL y pgAdmin
1. **Acceder a PostgreSQL y pgAdmin:**
   - Abre tu navegador y accede a las siguientes direcciones:
     - PostgreSQL: `localhost:5432`
     - pgAdmin: `localhost:8082`
   - Utiliza las credenciales definidas en el archivo `docker-compose.yaml`.

Paso 8: Configuración Inicial de PostgreSQL y pgAdmin
1. **Crear Usuario y Base de Datos en PostgreSQL:**
   - Utiliza pgAdmin para conectarte al servidor PostgreSQL y crea un nuevo usuario y una nueva base de datos.

2. **Iniciar Sesión en pgAdmin:**
   - Inicia sesión en pgAdmin con las credenciales proporcionadas en el archivo `docker-compose.yaml`.
   - Agrega un nuevo servidor PostgreSQL utilizando la dirección `postgres`.

3. **Configurar pgAdmin para Conectar con PostgreSQL:**
   - Configura pgAdmin para conectarse al servidor PostgreSQL en Docker.

Paso 9: Limpieza
1. **Detener y Eliminar Contenedores:**
   - Para detener los servicios, ejecuta:
     ```bash
     docker-compose down
     ```
   - Limpia los recursos una vez que hayas terminado la práctica.

Conclusión:
- Has completado la instalación y configuración de PostgreSQL y pgAdmin utilizando Docker y Docker Compose. Ahora puedes utilizar este entorno para desarrollar y administrar bases de datos de manera eficiente.