UNIVERSIDAD DEL VALLE

FACULTAD DE INGENIERÍA

Configuración de Base de Datos con Docker, PostgreSQL y pgAdmin.

BASES DE DATOS

Docente: Jefferson Amado Peña Torres

David Taborda Montenegro
202242264 - 3743
Ingeniería de Sistemas

Santiago de Cali

Octubre de 2025

Introducción

El presente informe tiene como propósito documentar de manera formal y técnica el proceso de configuración de un entorno de base de datos utilizando contenedores Docker para desplegar PostgreSQL como motor principal y pgAdmin 4 como herramienta gráfica de administración. El entorno fue preparado siguiendo un enfoque reproducible y aislado, con el objetivo de cumplir con los requerimientos del curso de Bases de Datos y garantizar buenas prácticas profesionales en la gestión de servicios.

Se abordará la instalación y configuración de WSL2, Docker Desktop, los contenedores de PostgreSQL y pgAdmin, la conexión entre ellos, y finalmente la creación y poblamiento de la base de datos servilimar con datos coherentes y realistas.

Objetivos

- Desplegar un contenedor con PostgreSQL utilizando Docker.
- Desplegar un contenedor independiente con pgAdmin 4.
- Garantizar la comunicación entre ambos contenedores desde Linux (WSL2).
- Conectar pgAdmin correctamente al contenedor de PostgreSQL.
- Crear las tablas y poblar la base de datos con datos realistas mediante DDL y DML.

Entorno de Trabajo

El entorno se implementó sobre un sistema operativo anfitrión Windows 11 Pro, utilizando Docker Desktop con integración habilitada para WSL2 (Windows Subsystem for Linux), específicamente con la distribución Ubuntu. Esta configuración permitió ejecutar contenedores Linux de forma nativa sobre el ecosistema Windows, garantizando un entorno aislado, reproducible y adecuado para despliegues profesionales.

Como motor de base de datos se utilizó PostgreSQL 14, desplegado mediante contenedor Docker, mientras que la administración gráfica se realizó a través de pgAdmin 4, también ejecutado dentro de un contenedor independiente. La base de datos configurada recibió el nombre servilimar, y el usuario principal utilizado para la conexión fue ulimar, con la contraseña ex4men_db.

Preparación del Entorno

1. Instalación de WSL2:
    - Se habilitaron las características de Windows Subsistema de Windows para Linux y Plataforma de máquina virtual.
    - Se instaló Ubuntu desde PowerShell.
    - Se verificó la versión de WSL ejecutando: wsl –list –verbose

2. Instalación y verificación de Docker Desktop:
    - Se descargó Docker Desktop para Windows.
    - Se habilitó la integración con WSL2 en la configuración de Docker.
    - Verificación de Docker ejecutando: docker –version y docker

Inicio de Contenedor PostgreSQL

docker run –name servilimar -e POSTGRES_USER=ulimar -e POSTGRES_PASSWORD=ex4men_db -p 5432:5432 postgres: 14

Inicio de Contenedor pgAdmin

docker run --rm -p 5050:80 -e "PGADMIN_DEFAULT_EMAIL=usuario@servilimar.com" -e "PGADMIN_DEFAULT_PASSWORD=limar#123" dpage/pgadmin4

Conexión desde pgAdmin a PostgreSQL

Se utilizó la IP interna del contenedor PostgreSQL (172.17.0.2) para establecer la conexión en pgAdmin: - Host: 172.17.0.2 - Puerto: 5432 - Usuario: ulimar - Contraseña: ex4men_db - Base de datos: servilimar

Creación de Tablas (DDL)

Tal como se establece en el video, se crearon las tablas pertinentes para cada una de las entidades involucradas en el enunciado. Así mismo, se desarrollaron algunos registros para dichas tablas, los cuales fueron exitosamente rellenados.

Aquí algunos ejemplos de ambas partes:

```sql
-- DESARROLLO PARTE PRÁCTICA EXAMEN PARCIAL.

CREATE TABLE usuario(
    usuario_id INTEGER PRIMARY KEY,
    nombre VARCHAR(100),
    apellido VARCHAR(100),
    tipo_usuario VARCHAR(100),
    condicion_especial VARCHAR(100),
    ciudad_id INTEGER,
    FOREIGN KEY (ciudad_id) REFERENCES ciudad(ciudad_id)
);

-- Tabla de Empleado, quien es una especialización de Usuario.
CREATE TABLE empleado(
    empleado_id INT PRIMARY KEY,
    usuario_id INT,
    FOREIGN KEY (usuario_id) REFERENCES usuario(usuario_id)
);

CREATE TABLE cargo(
    cargo_id INTEGER PRIMARY KEY,
    nombre_cargo VARCHAR(100),
    empleado_id INTEGER,
    FOREIGN KEY (empleado_id) REFERENCES empleado(empleado_id)
);

CREATE TABLE departamento(
    departamento_id INT PRIMARY KEY,
    nombre_departamento VARCHAR(100),
    empleado_id INT,
    FOREIGN KEY (empleado_id) REFERENCES empleado(empleado_id)
);
```

```sql
INSERT INTO empleado (empleado_id, usuario_id) VALUES
                                        ( empleado_id 301,  usuario_id 202),  -- María (Empleado)
                                        ( empleado_id 302,  usuario_id 204),  -- Daniela (Empleado)
                                        ( empleado_id 303,  usuario_id 206),  -- Laura (Empleado)
                                        ( empleado_id 304,  usuario_id 208),  -- Camila (Empleado)
                                        ( empleado_id 305,  usuario_id 210),  -- Valentina (Empleado)
                                        ( empleado_id 306,  usuario_id 201),  -- Juan (Cliente, lo metemos como empleado eventual)
                                        ( empleado_id 307,  usuario_id 203),  -- Carlos
                                        ( empleado_id 308,  usuario_id 205),  -- Andrés
                                        ( empleado_id 309,  usuario_id 207),  -- Felipe
                                        ( empleado_id 310,  usuario_id 209);  -- Sebastián
INSERT INTO cargo (cargo_id, nombre_cargo, empleado_id) VALUES
                                            ( cargo_id 401,  nombre_cargo 'Administrador de plataforma',  empleado_id 301),
                                            ( cargo_id 402,  nombre_cargo 'Analista de soporte',  empleado_id 301),

                                            ( cargo_id 403,  nombre_cargo 'Coordinador de operaciones',  empleado_id 302),
                                            ( cargo_id 404,  nombre_cargo 'Asistente de servicio',  empleado_id 302),

                                            ( cargo_id 405,  nombre_cargo 'Jefe de talento humano',  empleado_id 303),

                                            ( cargo_id 406,  nombre_cargo 'Analista junior',  empleado_id 304),
                                            ( cargo_id 407,  nombre_cargo 'Apoyo operativo',  empleado_id 304),

                                            ( cargo_id 408,  nombre_cargo 'Gerente regional',  empleado_id 305),

                                            ( cargo_id 409,  nombre_cargo 'Líder de turnos',  empleado_id 306),
                                            ( cargo_id 410,  nombre_cargo 'Auxiliar administrativo',  empleado_id 307);
```

Verificación y Resultados

- Se verificó la conexión desde pgAdmin, visualizando las tablas y registros correctamente.
- No se detectaron errores de datos ni inconsistencias.
- El entorno permitió manipular datos y realizar consultas sin problemas.

Conclusiones

- Docker + WSL2 proporcionan un entorno aislado y reproducible, ideal para pruebas y prácticas académicas de bases de datos.
- La separación de contenedores entre PostgreSQL y pgAdmin facilita la administración y el despliegue.
- La principal dificultad fue la conexión entre contenedores y pgAdmin, resuelta usando la IP interna del contenedor.
- Para futuras sesiones, se recomienda automatizar la creación de contenedores mediante docker-compose y versionar scripts DDL/DML para agilizar el despliegue.

Enlaces a video y GitHub:

Video: https://youtu.be/uloQ9TQUGGk?si=4hHHr9eCd9K9oz1a

GitHub: https://github.com/davidtaborda2025/Parte-Practica-Examen-Parcial-Bases-de-Datos.git