



# E3 - Contenedores en Red: Adminer y MariaDB

Álvaro Ronco Acebal y Guillermo Jesús Martín Pérez

## Introducción

En este ejercicio, se desplegarán dos contenedores en una red Docker:

- Un contenedor con **MariaDB** como servidor de base de datos.
- Un contenedor con **Adminer** (o phpMyAdmin) como interfaz gráfica para gestionar la base de datos.

La clave de este ejercicio es conectar ambos contenedores a través de una red bridge personalizada en Docker.

## Crear una red personalizada en Docker

Para que los contenedores puedan comunicarse, creamos una red llamada `redbdd`:

```
$docker network create redbdd
```

```
alvaro@alvaro-VirtualBox ~$ docker network create redbdd
7accf7c510ba1d6807d4fc2c100726262f5619d655a5749c87beaf9f646b7d63
```

Para ver los parámetros de la red creada podemos usar el siguiente comando:

```
$docker network inspect
```

```
alvaro@alvaro-VirtualBox ~$ docker network inspect redbdd
[
  {
    "Name": "reddbd",
    "Id": "b1b178de79a1d8e978eb39bc064bf09830f413a6547781392f8f56b6dad66fc3",
    "Created": "2025-02-13T09:13:35.372912522+01:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.20.0.0/16",
          "Gateway": "172.20.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "b32a68b9c70989ab626914829241e382a457e8af21ee94c9e0805c6dfd20e1a1": {
        "Name": "adminer",
        "EndpointID": "d87f8124dee8c7b0ee3605976bc4b577865aab51baa64cba66d5ea274797f678",
        "MacAddress": "02:42:ac:14:00:02",
        "IPv4Address": "172.20.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]
```

## Crear el contenedor de MariaDB

Ejecutamos un contenedor con la imagen oficial de MariaDB dentro de la red `reddbd`:

```
$docker run -d --name mariadb --network redbdd -e MYSQL_ROOT_PASSWORD=root -e
MYSQL_DATABASE=base -e MYSQL_USER=daw -e MYSQL_PASSWORD=daw -v mariadb-
data:/var/lib/mysql mariadb
```

Explicación de los parámetros:

- `-name mariadb` → Nombre del contenedor.
- `-network redbdd` → Conecta el contenedor a la red `reddbd`.
- `-e MYSQL_ROOT_PASSWORD=root` → Establece la contraseña del root.
- `-e MYSQL_DATABASE=base` → Crea la base de datos `base`.
- `-e MYSQL_USER=daw` y `-e MYSQL_PASSWORD=daw` → Crea el usuario `daw`.
- `-v mariadb-data:/var/lib/mysql` → Utiliza un volumen persistente.

```
alvaro@alvaro-VirtualBox ~$ docker run -d --name mariadb --network redbdd -e MYS
QL_ROOT_PASSWORD=root -e MYSQL_DATABASE=base -e MYSQL_USER=daw -e MYSQL_PASSWORD
=daw -v mariadb-data:/var/lib/mysql mariadb
322f335b8c275b78c742fb60b71d068f6351e6e29f0e9b99c4857537a74b243e
```

## Crear el contenedor de Adminer

Adminer es una interfaz web que permite gestionar bases de datos de forma gráfica. Se ejecuta con el siguiente comando:

```
$docker run -d --name adminer --network redbdd -p 8080:8080 adminer
```

Explicación de los parámetros:

- `--network redbdd` → Conecta Adminer a la misma red que MariaDB.
- `-p 8080:8080` → Expone Adminer en el puerto 8080.

```
alvaro@alvaro-VirtualBox ~$ docker run -d --name adminer --network redbdd -p 8080:8080 adminer
Unable to find image 'adminer:latest' locally
latest: Pulling from library/adminer
73226dab8db5: Pull complete
ed94e1c95a57: Pull complete
884bce373183: Pull complete
9a4cd7b75371: Pull complete
574dfab7cda2: Pull complete
798a45c9628c: Pull complete
c82cd9b427d9: Pull complete
Digest: sha256:34d37131366c5aa84e1693dbed48593ed6f95fb450b576c1a7a59d3a9c9e8802
Status: Downloaded newer image for adminer:latest
3684bd5b8655bbf69f669930cb1e4fc98c8f67ed82988fd108386be78dae87b5
```

Para comprobar que ambos contenedores estan funcionando correctamente podemos usar:

```
$docker ps
```

```
alvaro@alvaro-VirtualBox ~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
b32a68b9c709   adminer   "entrypoint.sh php -..." 3 seconds ago Up 2 seconds  0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   adminer
alvaro@alvaro-VirtualBox ~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
b32a68b9c709   adminer   "entrypoint.sh php -..." About a minute ago Up About a minute  0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   adminer
09591906adea   mariadb   "docker-entrypoint.s..." 2 minutes ago Created                                      mariadb
```

## Acceder a Adminer

Para acceder a “adminer” abrimos un navegador y vamos a la siguiente url:

```
http://localhost:8080
```

Idioma: Español

Adminer 4.8.1

## Login

Motor de base de datos	MySQL
Servidor	db
Usuario	
Contraseña	
Base de datos	

Login ☐ Guardar contraseña

Una vez en la pestaña anterior introducimos las credenciales de acceso creadas al crear el contenedor de mariadb:

- **Servidor:** mariadb (el nombre del contenedor)
- **Usuario:** daw
- **Contraseña:** daw
- **Base de datos:** base

Idioma: Español

MySQL » mariadb » Base de datos: base

Cerrar sesión

Adminer 4.8.1

DB: base

Comando SQL Importar Exportar Crear tabla

No existen tablas.

## Base de datos: base

Modificar Base de datos Esquema de base de datos Privilegios

### Tablas y vistas

No existen tablas.

Crear tabla Crear vista

### Procedimientos

Crear procedimiento Crear función

### Eventos

Crear Evento

## Crear una base de datos y una tabla desde Adminer

Dentro de Adminer, crea una nueva tabla con el siguiente SQL:

```
CREATE TABLE empleados (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  nombre VARCHAR(100),  
  puesto VARCHAR(50)  
);
```

```
CREATE TABLE empleados (  
  id INT AUTO INCREMENT PRIMARY KEY,  
  nombre VARCHAR(100),  
  puesto VARCHAR(50)  
)
```

Consulta ejecutada, 0 registros afectados. (0.041 s) [Modificar](#)

```
CREATE TABLE empleados (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  nombre VARCHAR(100),  
  puesto VARCHAR(50)  
);
```

## Eliminar los contenedores, la red y los volúmenes

Cuando terminemos de usar los contenedores, podemos eliminarlos con:

```
$docker stop mariadb adminer  
$docker rm mariadb adminer  
$docker network rm redbdd  
$docker volume rm mariadb-data
```

```
alvaro@alvaro-VirtualBox ~$ docker stop mariadb adminer  
mariadb  
adminer  
alvaro@alvaro-VirtualBox ~$ docker rm mariadb adminer  
mariadb  
adminer  
alvaro@alvaro-VirtualBox ~$ docker network rm redbdd  
redbdd  
alvaro@alvaro-VirtualBox ~$ docker volume rm mariadb-data  
mariadb-data
```