```
version: "3.9"
services:
  db:
     image: mysql:5.7
     volumes:
        - db_data:/var/lib/mysql
     restart: always
     environment:
        MYSQL_ROOT_PASSWORD: somewordpress
        MYSQL_DATABASE: wordpress
        MYSQL_USER: wordpress
        MYSQL_PASSWORD: wordpress
  wordpress:
     depends_on:
        - db
     image: wordpress:latest
     ports:
        - "8000:80"
     restart: always
     environment:
        WORDPRESS_DB_HOST: db:3306
        WORDPRESS_DB_USER: wordpress
        WORDPRESS_DB_PASSWORD: wordpress
        WORDPRESS_DB_NAME: wordpress
volumes:
  db_data:
```

SERVICES

```
     services:
       db:
          image: mysql:5.7
          volumes:
             - db_data:/var/lib/mysql
          restart: always
          environment:
             MYSQL_ROOT_PASSWORD: somewordpress
             MYSQL_DATABASE: wordpress
             MYSQL_USER: wordpress
             MYSQL_PASSWORD: wordpress
```

En primer lugar, establece la variable “**services**”. Siguiendo el formato **YAML**, lo que almacena “**services**” es un array asociativo con las variables “**db**” y posteriormente “**wordpress**” (no se ve en este fragmento). Para saber que un elemento está

contenido en otro, lo detecta mediante el nivel de los espacios (de forma similar a lenguajes como **Python**).

Cada una de esas variables, define un contenedor. En este caso, la plantilla de contenedor "**db**", es también un array asociativo en formato **YAML** y define las siguientes propiedades:

```
image: mysql:5.7
```

Define la imagen que utilizará el contenedor.

```
volumes:
   - db_data:/var/lib/mysql
```

Define qué volúmenes contendrá esta imagen. En este caso, los volúmenes en sí, se definen como una lista, de un solo elemento. Sabemos que es una lista porque en YAML empiezan con "-".

En este caso, se define el volumen "db_data" que se mapea con el directorio del contenedor "/var/lib/mysql". Si quisiéramos tener varios volúmenes, podríamos hacer algo similar a:

```
volumes:
   - db_data:/var/lib/mysql
   - otrovolumen:/directorio/otrovolumen
```

Con esto ya hemos visto los 3 principales tipos elementos de YAML (par variable/valor, array asociativo, lista).

```
 restart: always
```

Este fragmento, usando la sintaxis YAML de par variable/valor, indica que si el servidor se detiene, "**Docker Compose**" lo re-lance automáticamente (útil para contenedores que puedan caer por un fallo, pero sean necesarios para que la aplicación funcione).

```
environment:
     MYSQL_ROOT_PASSWORD: somewordpress
     MYSQL_DATABASE: wordpress
     MYSQL_USER: wordpress
     MYSQL_PASSWORD: wordpress
```

En este último fragmento, se define la variable "**environment**" del contenedor. Estas contendrán mediante un vector asociativo, el par variable/valor de cada variable de

entorno definida en el contenedor. A efectos prácticos, está definiendo el password de root de **MySQL**, el nombre de una base de datos “wordpress”, un usuario con permisos de root para **MySQL** llamado “wordpress” (necesario para conexiones remotas) y su password como “wordpress”.

```
wordpress:
   depends_on:
      - db
   image: wordpress:latest
   ports:
      - "8000:80"
   restart: always
   environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
```

En este caso, se está definiendo dentro de services, la plantilla del contenedor “**wordpress**”.
Pasamos a revisar sus fragmentos:

```
depends_on:
   - db
```

Indica que este contenedor depende del contenedor “**db**”, por lo cual el contenedor “**db**” deberá ponerse en marcha antes de iniciar nuestro contenedor “**wordpress**”.

```
 image: wordpress:latest
```

Como se ha comentado anteriormente, indica la imágen en que se basa el contenedor.

```
ports:
   - "8000:80"
```

Indica que el puerto “**80**” de ese contenedor se mapea con el puerto “**8000**” del anfitrión.

```
restart: always
```

Como se ha comentado anteriormente, indica que si el contenedor se para, se reinicie automáticamente.

```
environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
```

De forma similar al anterior punto, definimos el valor de distintas “variables de entorno” del contenedor “wordpress”.

```
volumes:
  db_data:
```

Simplemente, indica que el volumen “db_data” está disponible para otros contenedores puestos en marcha con “Docker Compose”.