# **Capítulo 7 - Criando volumes para persistir dados**

Na nossa última aula vimos a interligação de redes entre os containers, aprendemos mais sobre como funciona a instrução **link** e como o Docker trata essa distribuição dinâmica dos ips entre os containers.

Sempre que criamos os containers da nossa aplicação, agora utilizando o *compose*, voltamos para a página inicial do WordPress para a escolha da língua, com o seguinte endereço: ip/wp-admin/install.php. Ou seja, tudo que estamos configurando ou escrevendo não está sendo persistido no banco de dados, no container do MySQL. Toda vez devemos passar pela instalação, preencher os campos de título, nome de usuário, etc. Veremos que existe uma forma dos dados serem salvos e persistidos.

Os dados que serão criados dentro do banco precisam ser persistidos no disco do Host Local. Para exemplificarmos, vamos criar um container normalmente, porém passando o diretório local do "docker-compose.yml" com a instrução **-v** para que o arquivo seja mapeado para dentro do container:

docker run -it -v $(pwd):/tmp ubuntu bash

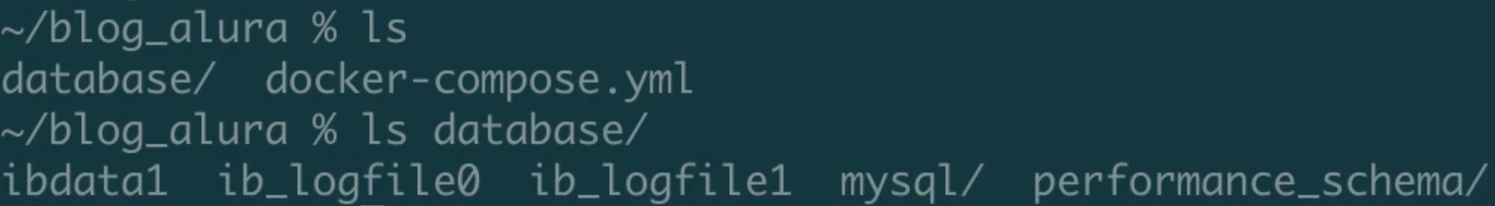
O **-v** cria um volume para dentro do container. Todos os arquivos que forem criados a partir dessa conexão entre nosso Host Local e o container serão persistidos. Perceba que os arquivos serão criados dentro do container:



Mesmo destruindo todos os containers, os arquivos permanecem. Podemos aplicar essa lógica, então, dentro do arquivo de compose

db:  
 image: mysql  
 volumes:  
 - ~/blog\_alura/database/:/var/lib/mysql/  
...

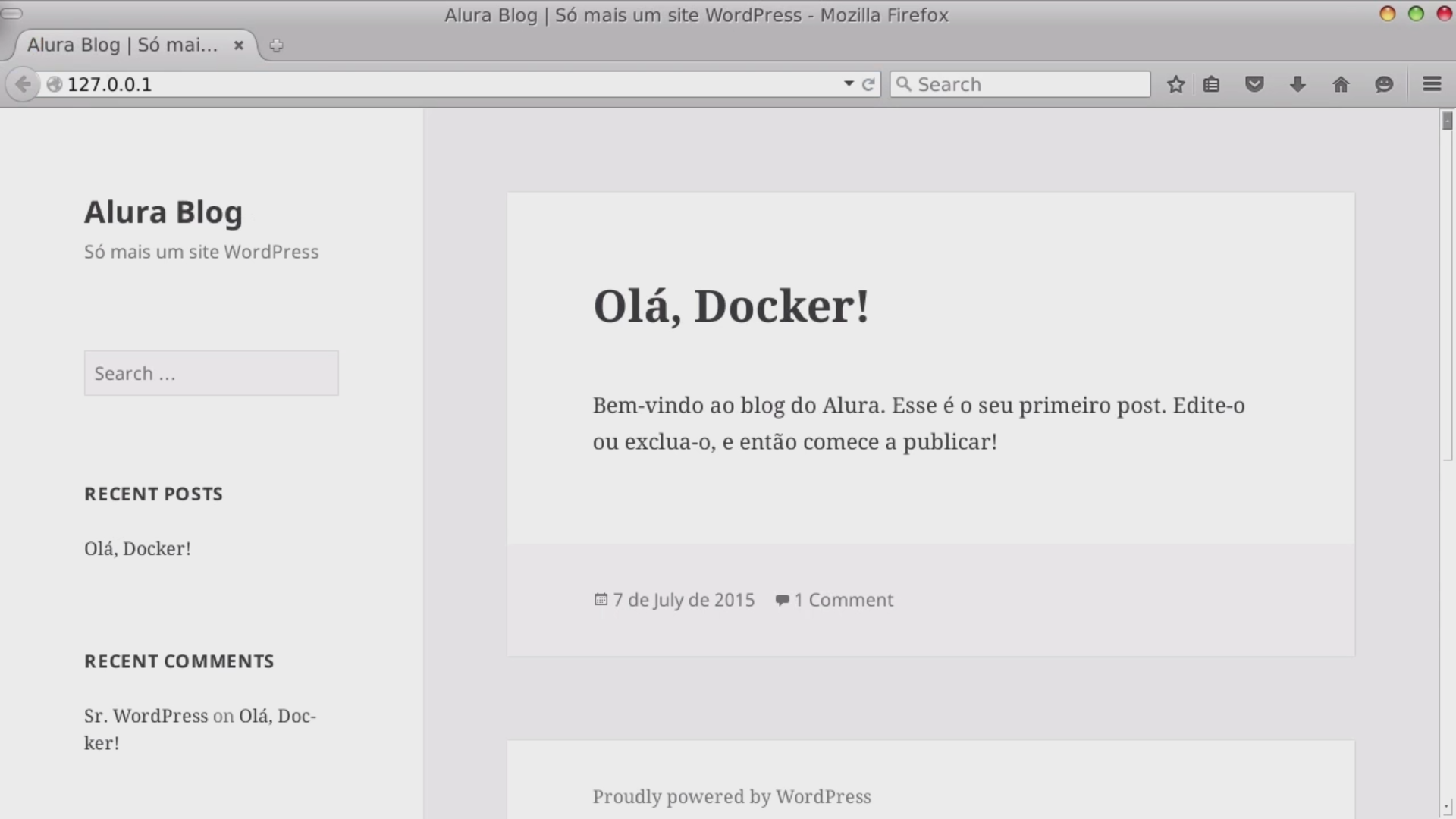
Na declaração para **volumes** passamos o diretório que será mapeado com o do MySQL, onde os dados serão salvos (var/lib/mysql). Assim, quando fizermos **docker-compose up -d**, serão criados os containers já fazendo a estrutura de mapeamento trazendo os diretórios e arquivos os quais o MySQL utiliza para persistirem no banco. E, de fato, após rodado o compose:



Testemos novamente, fazendo a instalação do WordPress e passando pelas configurações necessárias. Voltamos para o Terminal e matamos e removemos os containers:

docker-compose kill  
  
docker-compose rm

Não importa. Os arquivos e diretórios serão reutilizados assim que subirmos novamente o compose. Os dados agora são persistentes através de volumes. O nosso blog será inicializado onde paramos. Podemos fazer alterações que elas também serão salvas:



Com a instrução **volumes** nós conseguimos fazer com que os dados criados através do MySQL realmente existam em um determinado diretório na nossa máquina local.