






## 04. Persistindo dados

 Date	@23/02/2023
 Categoria	Docker
 Curso	Docker: criando e gerenciando containers

### Tópicos

- O problema de persistir dados
- Tipos de persistência
- Utilizando bind mounts
- Criando um bind
- Utilizando volumes
- Vantagens de volumes
- Faça como eu fiz: Criando um volume
- Utilizando tmpfs
- O que aprendemos?

### Tipos de persistência

Podemos querer que os dados da nossa aplicação sejam persistentes, porque assim garantimos que ela esteja distribuída e disponível se precisarmos consultá-la. Porém, se escrevermos os dados nos containers, por padrão eles não ficarão armazenados nesta camada, criada para ser descartável. Existem três possibilidades para contornar esta situação com o Docker.

Selecione as alternativas com meios para persistir dados importantes.

- StorageBuilder.
- Volumes.

- Com volumes, é possível escrever os dados em uma camada persistente.
  - Bind mounts.
    - com bind mounts, é possível escrever os dados em uma camada persistente baseado na estrutura de pastas do host
  - StorageMaker.
- 

## Criando um bind

Anteriormente entendemos que os bind mounts são capazes de persistir dados de containers através de um vínculo criado com a estrutura de pastas do nosso host. Porém, ainda precisamos fixar como criar um.

Qual das alternativas abaixo contém a sintaxe correta para a criação de um bind mount?

- `docker run -mount type=bind nginx`
  - `docker run -mount type=bind,source=/home/diretorio,target=/app nginx`
  - `docker run -mount source=/home/diretorio,target=/app nginx`
- 

## Vantagens de volumes

Já aprendemos sobre a possibilidade de usar bind mounts e volumes para persistir dados no ambiente Docker. Porém, com a utilização de volumes, temos uma vantagem em relação aos bind mounts.

Escolha a alternativa que apresenta a vantagem do uso de volumes.

- **Volumes são gerenciados pelo Docker e independem da estrutura de pastas do sistema.**
  - Desta maneira, a persistência de dados independe de como as pastas do sistema estão estruturadas.
- Volumes possuem maior capacidade de armazenamento.
  - A capacidade não é influenciada pelo tipo de armazenamento.
- Volumes acessam os dados de maneira mais rápida.

- Não existe relação entre o tipo de armazenamento e a velocidade de acessar os dados.

---

## Faça como eu fiz: Criando um volume

Agora criaremos nosso primeiro volume gerenciado pelo Docker, com o objetivo de garantir que nossos dados persistam mesmo sem uma preocupação com a estrutura de diretórios de nosso sistema. Dessa forma será possível armazenar e reutilizar arquivos entre execuções de containers, o que é muito útil para aplicações stateful.

Inicialmente, abra o terminal e execute o comando `docker volume ls` e veja que a saída está vazia, pois ainda não criamos nenhum volume até então.

Em seguida, execute o comando `docker volume create meu-volume` e execute novamente o comando `docker volume ls`. Veja que desta vez o nosso recém-criado volume está sendo exibido.

Com o volume criado, agora iremos executar um novo container vinculado ao volume. Para isso, execute o comando `docker run -it -v meu-volume:/app ubuntu bash`. Dentro do container, execute o comando `cd /app` para acessar o diretório e crie um arquivo com o comando `um-arquivo-qualquer`. Saia do container com o comando `Ctrl D` e execute mais uma vez o comando `docker run -it -v meu-volume:/app ubuntu bash`. Por fim, execute o comando `ls /app` e veja que o arquivo `um-arquivo-qualquer` criado anteriormente continua presente.

---

## O que aprendemos?

Nessa aula aprendemos:

- Quando containers são removidos, nossos dados são perdidos;
- Podemos persistir dados em definitivo através de volumes e bind mounts;
- Bind mounts dependem da estrutura de pastas do host;
- Volumes são gerenciados pelo Docker;
- Tmpfs armazenam dados em memória volátil.

