

## Sumário

Dockerfile .....	2
Criação e configuração do Dockerfile .....	2
Instalação e configuração de Docker-Compose (MySQL) .....	6
Arquivo docker-compose.yml .....	6
Arquivo init.sql .....	7
Inicializando o Docker-compose .....	7
Referências .....	8

## Dockerfile

O Docker pode construir imagens automaticamente lendo as instruções de um Dockerfile. Um Dockerfile é um documento de texto que contém todos os comandos que um usuário pode chamar na linha de comando para montar uma imagem.

A instrução não faz distinção entre maiúsculas e minúsculas. No entanto, a convenção é que eles estejam em MAIÚSCULAS para distingui-los dos argumentos com mais facilidade.

O Docker executa instruções em um Dockerfile em ordem. Um Dockerfile deve começar com uma instrução FROM. Isso pode ocorrer após diretivas do analisador, comentários e ARGs com escopo global. A instrução FROM especifica a imagem pai a partir da qual você está construindo. FROM só pode ser precedido por uma ou mais instruções ARG, que declaram argumentos que são usados em linhas FROM no Dockerfile.

O Docker trata as linhas que começam com # como um comentário, a menos que a linha seja uma diretiva de analisador válida. Um marcador # em qualquer outro lugar de uma linha é tratado como um argumento.

### Criação e configuração do Dockerfile

Crie um diretório para os arquivos SQL: Crie um diretório chamado "arquivos\_sql" no mesmo diretório onde você criou o Dockerfile. Coloque os scripts SQL que configuram suas tabelas nesse diretório.

```
mkdir arquivos_sql  
cd arquivos_sql  
touch 001-tabelas.sql 002-dados.sql
```

No arquivo **001-tabelas.sql**

```
CREATE DATABASE IF NOT EXISTS so_teste;  
  
USE so_teste;  
  
CREATE TABLE IF NOT EXISTS aluno(  
    id INT PRIMARY KEY AUTO_INCREMENT,  
    ra VARCHAR(20),  
    nome VARCHAR(100),  
    aniversario DATE  
)CHARACTER SET='utf8mb4';  
  
INSERT INTO aluno VALUES  
(null, '0123456', 'Eduardo Verri', '1991-06-16'),  
(null, '0254568', 'Marcio Santana', '1953-12-24'),  
(null, '0456587', 'Claudio Frizzarini', '1980-05-12');
```

E no arquivo **002-dados.sql**

```
USE so_teste;  
  
CREATE TABLE IF NOT EXISTS materia(  
    id INT PRIMARY KEY AUTO_INCREMENT,  
    materia VARCHAR(50)  
)CHARACTER SET = 'utf8mb4';  
  
INSERT INTO materia VALUES  
(null, 'sistemas operacionais'),  
(null, 'análise de sistemas'),  
(null, 'linguagem de programação');
```

Volte um nível com **cd ..** e crie um Dockerfile: Crie um arquivo chamado "Dockerfile" em um diretório de sua escolha. O Dockerfile será usado para construir a imagem do MySQL com as configurações e tabelas desejadas.

```
touch Dockerfile  
vim Dockerfile
```

```
# use a imagem oficial do MySQL como imagem base
FROM mysql:latest

# defina variáveis de ambiente para a senha do root do MySQL
ENV MYSQL_ROOT_PASSWORD=urubu100

# copie os scripts SQL de inicialização para um diretório temporário no container
COPY ./arquivos_sql/ /docker-entrypoint-initdb.d/

# exponha a porta padrão do MySQL
EXPOSE 3306
```

Se você tiver mais de um arquivo SQL na pasta arquivos\_sql e desejar executar todos eles durante a inicialização do contêiner, basta copiar todos os arquivos SQL relevantes para o diretório arquivos\_sql. Quando você construir a imagem e iniciar o contêiner, o MySQL executará todos os scripts SQL nesse diretório em ordem alfabética.

Por exemplo, se você tiver dois arquivos SQL, como init1.sql e init2.sql, o Docker copiará ambos os arquivos para o diretório docker-entrypoint-initdb.d/ no contêiner e os executará na ordem alfabética, ou seja, primeiro o init1.sql e depois o init2.sql.

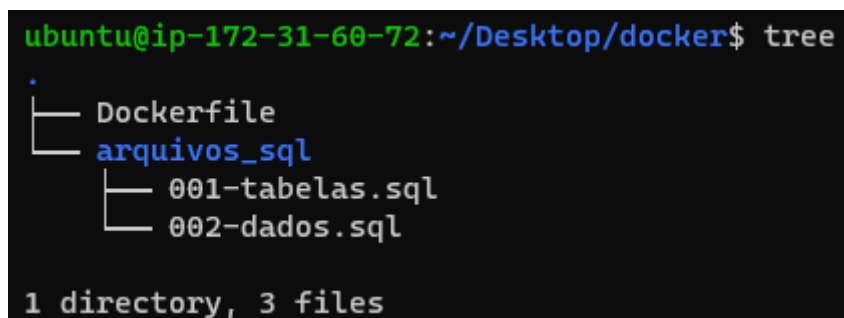
Certifique-se de nomear os arquivos SQL de forma que a ordem alfabética corresponda à ordem em que deseja que eles sejam executados, se a ordem for importante. Por exemplo, você pode nomeá-los como 001-init.sql, 002-outro-script.sql para garantir a ordem de execução desejada.

O diretório docker-entrypoint-initdb.d é um diretório especial em contêineres MySQL (e em outros contêineres relacionados ao MySQL, como o MariaDB) que permite a execução de scripts SQL durante a inicialização do contêiner. Qualquer script SQL colocado nesse diretório será executado automaticamente quando o contêiner MySQL for iniciado.

Isso é especialmente útil quando você deseja configurar tabelas, inserir dados ou realizar outras operações de inicialização personalizadas no banco de dados MySQL quando o contêiner é iniciado. É uma forma conveniente de automatizar o processo de configuração do banco de dados.

Os arquivos SQL colocados nesse diretório podem ter nomes personalizados, desde que tenham a extensão ".sql" e sigam uma ordem alfabética.

Ao final da configuração, a estrutura será parecida com:



```
ubuntu@ip-172-31-60-72:~/Desktop/docker$ tree
.
├── Dockerfile
└── arquivos_sql
    ├── 001-tabelas.sql
    └── 002-dados.sql

1 directory, 3 files
```

Agora só basta construir a imagem do Docker. Vá até o diretório onde criou o **Dockerfile** e execute o seguinte comando com as devidas permissões de super usuário:

```
docker build -t meu-banco .
```

Isso criará uma imagem Docker com o nome "meu-banco" usando o Dockerfile e os arquivos SQL no diretório. Agora é necessário executar o contêiner MySQL com as configurações e tabelas personalizadas com:

```
docker run -d --name meu-container -p 3306:3306 meu-banco
```

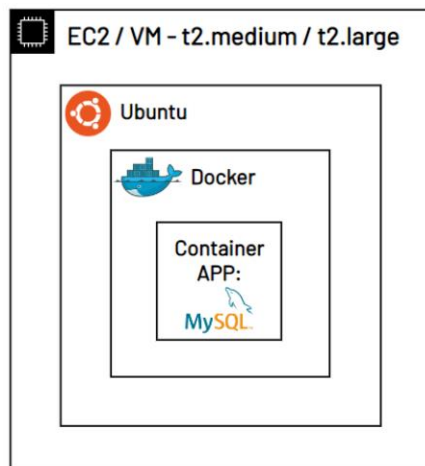
Isso iniciará um contêiner MySQL com as configurações definidas no Dockerfile e executará os scripts SQL no diretório "arquivos\_sql" para configurar suas tabelas.

Verifique o status do seu container com

**docker ps -a**

```
ubuntu@ip-172-31-60-72: ~/E X
ubuntu@ip-172-31-60-72:~/Desktop/docker$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
3189ec720c2a   meu-banco "docker-entrypoint.s...  17 minutes ago Up 17 minutes  0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   meu-cont
ainer
ubuntu@ip-172-31-60-72:~/Desktop/docker$
```

Lembrando da camada de acessos até o container de MySQL



```
ubuntu@ip-172-31-60-72:~/Desktop/docker$ sudo docker exec -it meu-container bash
bash-4.4# mysql -u root -purubu100
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> |
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| so_teste |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> use so_teste
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> |
```

```
mysql> show tables;
+-----+
| Tables_in_so_teste |
+-----+
| aluno               |
| materia             |
+-----+
2 rows in set (0.01 sec)

mysql> desc aluno;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id         | int           | NO   | PRI | NULL    | auto_increment |
| ra        | varchar(20)   | YES  |     | NULL    |                |
| nome      | varchar(100)  | YES  |     | NULL    |                |
| aniversario | date          | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> desc materia;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id        | int           | NO   | PRI | NULL    | auto_increment |
| materia   | varchar(50)   | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

```
mysql> select * from aluno;
+----+-----+-----+-----+
| id | ra      | nome                | aniversario |
+----+-----+-----+-----+
| 1  | 0123456 | Eduardo Verri      | 1991-06-16  |
| 2  | 0254568 | Marcio Santana     | 1953-12-24  |
| 3  | 0456587 | Claudio Frizzarini | 1980-05-12  |
+----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from materia;
+----+-----+
| id | materia                |
+----+-----+
| 1  | sistemas operacionais  |
| 2  | análise de sistemas    |
| 3  | linguagem de programação |
+----+-----+
3 rows in set (0.00 sec)
```

## Instalação e configuração de Docker-Compose [MySQL]

Docker-Compose é uma ferramenta para definir e executar aplicativos Docker com vários contêineres. Com o Compose, você usa um arquivo YAML para configurar os serviços do seu aplicativo. Então, com um único comando, você cria e inicia todos os serviços da sua configuração.

No terminal atualize os pacotes antes de instalar o Docker-compose

```
sudo apt update
```

E então

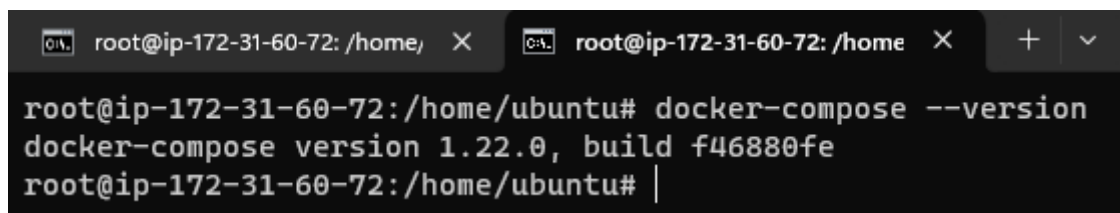
```
sudo curl -L "https://github.com/docker/compose/releases/download/v2.23.0/docker-  
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose && sudo chmod +x  
/usr/local/bin/docker-compose
```

ou

```
curl -SL https://github.com/docker/compose/releases/download/v2.23.0/docker-  
compose-linux-x86_64 -o /usr/local/bin/docker-compose
```

Após a instalação pode checar a versão do Docker-compose com

```
docker-compose -version
```



```
root@ip-172-31-60-72: /home/ X root@ip-172-31-60-72: /home X + v  
root@ip-172-31-60-72:/home/ubuntu# docker-compose --version  
docker-compose version 1.22.0, build f46880fe  
root@ip-172-31-60-72:/home/ubuntu# |
```

Crie um diretório para guardar teus arquivos YAML e os arquivos de script SQL.

Crie uma estrutura de pasta como abaixo, o script init.sql será a configuração do banco de dados. Enquanto, que o docker-compose.yml configurará as informações do container

```
|  
├── docker-compose.yml  
├── init-scripts  
│   └── init.sql
```

Arquivo docker-compose.yml

```
version: '3.3'
```

```
services:  
  mysql:  
    container_name: testeSO  
    image: mysql:5.7  
    restart: always  
    environment:  
      MYSQL_ROOT_PASSWORD: urubu100  
      MYSQL_DATABASE: aulaSO  
    volumes:  
      - ./init-scripts:/docker-entrypoint-initdb.d  
    ports:  
      - "3306:3306"
```

```
volumes:  
  mysql_data:
```

## Arquivo init.sql

```
-- Arquivo init.sql
CREATE TABLE cliente (
  id INT AUTO_INCREMENT,
  nome_cliente VARCHAR(255),
  PRIMARY KEY (id)
);
```

## Inicializando o Docker-compose

No terminal digite

**docker-compose up**

```
root@ip-172-31-60-72: /home X root@ip-172-31-60-72: /home/ X + v
testeSO | 2023-10-25T18:16:33.734647Z | [Note] InnoDB: Compressed tables use zlib 1.2.13
testeSO | 2023-10-25T18:16:33.734651Z | [Note] InnoDB: Using Linux native AIO
testeSO | 2023-10-25T18:16:33.735340Z | [Note] InnoDB: Number of pools: 1
testeSO | 2023-10-25T18:16:33.735709Z | [Note] InnoDB: Using CPU crc32 instructions
testeSO | 2023-10-25T18:16:33.738149Z | [Note] InnoDB: Initializing buffer pool, total size = 128M, instances = 1, chunk size = 128M
testeSO | 2023-10-25T18:16:33.748470Z | [Note] InnoDB: Completed initialization of buffer pool
testeSO | 2023-10-25T18:16:33.751713Z | [Note] InnoDB: If the mysqld execution user is authorized, page cleaner thread priority can be changed. See
the man page of setpriority().
testeSO | 2023-10-25T18:16:33.763816Z | [Note] InnoDB: Highest supported file format is Barracuda.
testeSO | 2023-10-25T18:16:33.779754Z | [Note] InnoDB: Creating shared tablespace for temporary tables
testeSO | 2023-10-25T18:16:33.779834Z | [Note] InnoDB: Setting file './ibtmp1' size to 12 MB. Physically writing the file full; Please wait ...
testeSO | 2023-10-25T18:16:33.820488Z | [Note] InnoDB: File './ibtmp1' size is now 12 MB.
testeSO | 2023-10-25T18:16:33.821908Z | [Note] InnoDB: 96 redo rollback segment(s) found. 96 redo rollback segment(s) are active.
testeSO | 2023-10-25T18:16:33.821929Z | [Note] InnoDB: 32 non-redo rollback segment(s) are active.
testeSO | 2023-10-25T18:16:33.822815Z | [Note] InnoDB: 5.7.43 started; log sequence number 12222291
testeSO | 2023-10-25T18:16:33.823150Z | [Note] InnoDB: Loading buffer pool(s) from /var/lib/mysql/ib_buffer_pool
testeSO | 2023-10-25T18:16:33.825307Z | [Note] Plugin 'FEDERATED' is disabled.
testeSO | 2023-10-25T18:16:33.827584Z | [Note] InnoDB: Buffer pool(s) load completed at 231025 18:16:33
testeSO | 2023-10-25T18:16:33.832897Z | [Note] Found ca.pem, server-cert.pem and server-key.pem in data directory. Trying to enable SSL support usi
ng them.
testeSO | 2023-10-25T18:16:33.832920Z | [Note] Skipping generation of SSL certificates as certificate files are present in data directory.
testeSO | 2023-10-25T18:16:33.832926Z | [Warning] A deprecated TLS version TLSv1 is enabled. Please use TLSv1.2 or higher.
testeSO | 2023-10-25T18:16:33.832929Z | [Warning] A deprecated TLS version TLSv1.1 is enabled. Please use TLSv1.2 or higher.
testeSO | 2023-10-25T18:16:33.834115Z | [Warning] CA certificate ca.pem is self signed.
testeSO | 2023-10-25T18:16:33.834174Z | [Note] Skipping generation of RSA key pair as key files are present in data directory.
testeSO | 2023-10-25T18:16:33.834685Z | [Note] Server hostname (bind-address): '*'; port: 3306
testeSO | 2023-10-25T18:16:33.834783Z | [Note] IPv6 is available.
testeSO | 2023-10-25T18:16:33.834803Z | [Note] - '::' resolves to '::';
testeSO | 2023-10-25T18:16:33.834830Z | [Note] Server socket created on IP: '::'.
testeSO | 2023-10-25T18:16:33.837455Z | [Warning] Insecure configuration for --pid-file: Location '/var/run/mysqld' in the path is accessible to al
l OS users. Consider choosing a different directory.
testeSO | 2023-10-25T18:16:33.847856Z | [Note] Event Scheduler: Loaded 0 events
testeSO | 2023-10-25T18:16:33.848278Z | [Note] mysqld: ready for connections.
testeSO | Version: '5.7.43' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server (GPL)
```

Caso queira rodar em background

**docker-compose up -d**

```
root@ip-172-31-60-72: /home X root@ip-172-31-60-72: /home/ X + v
root@ip-172-31-60-72: /home/ubuntu/compose-test# docker-compose up -d
Creating testeSO ... done
root@ip-172-31-60-72: /home/ubuntu/compose-test# |
```

Teste se foi criado e se está rodando o container

**docker ps -a**

```
root@ip-172-31-60-72: /home/ X root@ip-172-31-60-72: /home X + v
root@ip-172-31-60-72: /home/ubuntu/compose-test# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
7de1cd441f03   mysql:5.7     "docker-entrypoint.s..." 20 seconds ago Up 19 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp testeSO
root@ip-172-31-60-72: /home/ubuntu/compose-test#
```

## Referências

[Install Compose standalone | Docker Docs](#)

[How to Create a MySQL Instance with Docker Compose | by Chris Chuck | Medium](#)

[Dockerize MySQL with Docker-Compose | by Hamza Ak | Medium](#)

[Dockerfile reference | Docker Docs](#)