Crie um diretório para você:

```
$ mkdir visualops-files
```

mkdir visualops-files

Crie um docker-compose.yml e uma pasta para armazenar os scripts sql:

```
tigo@ip-172-31-61-112:~/visualops-files$ touch docker-compose.yml
```

```
tigo@ip-172-31-61-112:~/visualops-files$ mkdir sql-files
```

touch docker-compose.yml

mkdir sql-files

Clone o repositório shared_script e mova o seu jar individual para a pasta que você criou anteriormente:

```
tigo@ip-172-31-61-112:~/visualops-files$ git clone https://github.com/Organizacao-Grupo5/shared_script.
git
```

git clone https://github.com/Organizacao-Grupo5/shared_script.git

ou

git clone https://SEU_TOKEN_DE_ACESSO_DO_GITHUB@github.com/Organizacao-Grupo5/shared_script.git

link para criar token = https://github.com/settings/tokens

Deve ficar mais ou menos assim:

```
tigo@ip-172-31-61-112:~/visualops-files$ tree -L 1
.
├── docker-compose.yml
├── jar-individual-thiago
└── sql-files

3 directories, 1 file
```

Mude para o diretório de scripts sql e crie um arquivo para criar o usuário e as tabelas:

```
tigo@ip-172-31-61-112:~/visualops-files$ cd sql-files/
tigo@ip-172-31-61-112:~/visualops-files/sql-files$ touch create-script.sql
```

cd sql-files/

touch create-script.sql

Abra o arquivo com o Vim ou Nano (recomendo Vim pois não sei usar o Nano):

```
-- SQLBook: Code
DROP DATABASE IF EXISTS visualops_db;
CREATE DATABASE visualops_db;

CREATE USER IF NOT EXISTS 'client'@'%' IDENTIFIED BY 'Client123$';
GRANT SELECT, UPDATE, DELETE, INSERT ON visualops_db.* TO 'client';
FLUSH privileges;

USE visualops_db;
```


```
-- SQLBook: Code
DROP DATABASE IF EXISTS visualops_db;
CREATE DATABASE visualops_db;

CREATE USER IF NOT EXISTS 'client'@'%' IDENTIFIED BY 'Client123$';
GRANT SELECT, UPDATE, DELETE, INSERT ON visualops_db.* TO 'client';
FLUSH privileges;

USE visualops_db;
```


use essa configuração antes de criar as tabelas

Cole seu script de banco de dados:

```
-- SQLBook: Code
DROP DATABASE IF EXISTS visualops_db;
CREATE DATABASE visualops_db;

CREATE USER IF NOT EXISTS 'client'@'%' IDENTIFIED BY 'Client123$';
GRANT SELECT, UPDATE, DELETE, INSERT ON visualops_db.* TO 'client';
FLUSH privileges;

USE visualops_db;

CREATE TABLE plano (
    idPlano INT PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(30) NOT NULL,
    descricao VARCHAR(200) NOT NULL
);

CREATE TABLE empresa (
    idEmpresa INT PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(50),
    cnpj CHAR(18),
    fkPlano INT,
    CONSTRAINT fkPlano FOREIGN KEY (fkPlano) REFERENCES plano(idPlano)
);

CREATE TABLE endereco (
    idEndereco INT PRIMARY KEY AUTO_INCREMENT,
    cep CHAR(9) NOT NULL,
    logradouro VARCHAR(60) NOT NULL,
    numero VARCHAR(4) NOT NULL,
    bairro VARCHAR(40),
    estado VARCHAR(30),
    complemento VARCHAR(45),
    fkEmpresa INT NOT NULL,
        CONSTRAINT fkEmpresaEndereco FOREIGN KEY (fkEmpresa) REFERENCES empresa(idEmpresa)
);

CREATE TABLE usuario (
    idUsuario INT PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(60) NOT NULL,
    email VARCHAR(60) NOT NULL UNIQUE,
    senha VARCHAR(45),
    cargo varchar(45),
    fkEmpresa INT NOT NULL,
        CONSTRAINT fkEmpresaUsuario FOREIGN KEY (fkEmpresa) REFERENCES empresa(idEmpresa)

);
```

exemplo do meu. (o meu está com algumas alterações que não tem no em grupo pois isso não mandei o script inteiro)

Observação: em tabelas que tem foreign Keys compostas deve-se colocar tudo dentro de uma constraint como mostrado na imagem abaixo:

```
CREATE TABLE componente (
    idComponente INT AUTO_INCREMENT,
    componente VARCHAR(200) NOT NULL,
    modelo VARCHAR(200) NOT NULL,
    fabricante VARCHAR(200) NOT NULL,
    fkMaquina INT NOT NULL,
    fkUsuario INT NOT NULL,
    PRIMARY KEY (idComponente, fkMaquina, fkUsuario),
    CONSTRAINT fkMaquinaComponente FOREIGN KEY (fkMaquina, fkUsuario) REFERENCES maquina(idMaquina, fkU
suario)
);
```

Ao finalizar as mudanças no sql, volte para a pasta que você criou:

```
tigo@ip-172-31-61-112:~/visualops-files$ cd sql-files/
tigo@ip-172-31-61-112:~/visualops-files/sql-files$ cd ../
tigo@ip-172-31-61-112:~/visualops-files$ ls
docker-compose.yml  jar-individual-thiago  sql-files
tigo@ip-172-31-61-112:~/visualops-files$
```

Agora mova seu jar individual para a pasta atual:

```
tigo@ip-172-31-61-112:~/visualops-files$ mv '/shared_scripts/Jar Individuais/jar-individual-...' .
```

mv ‘/shared_scripts/Jar Individuais/jar-individual-SEU_NOME’ .

Agora já está nesse jeito:

```
tigo@ip-172-31-61-112:~/visualops-files$ ls
docker-compose.yml  jar-individual-thiago  sql-files
```

Em seguida entre no seu jar e crie o Dockerfile:

```
tigo@ip-172-31-61-112:~/visualops-files$ cd jar-individual-thiago/
tigo@ip-172-31-61-112:~/visualops-files/jar-individual-thiago$ cat Dockerfile
FROM maven:3.8.5-openjdk-17 AS builder

WORKDIR /app

COPY . /app

RUN mvn clean install

FROM openjdk:23-ea-17-jdk

COPY --from=builder /app/target/jar-individual-thiago-1.0-SNAPSHOT-jar-with-dependencies.jar /app.jar

CMD ["java", "-jar", "/app.jar"]
```

cd jar-individual-SEU_NOME

vim Dockerfile

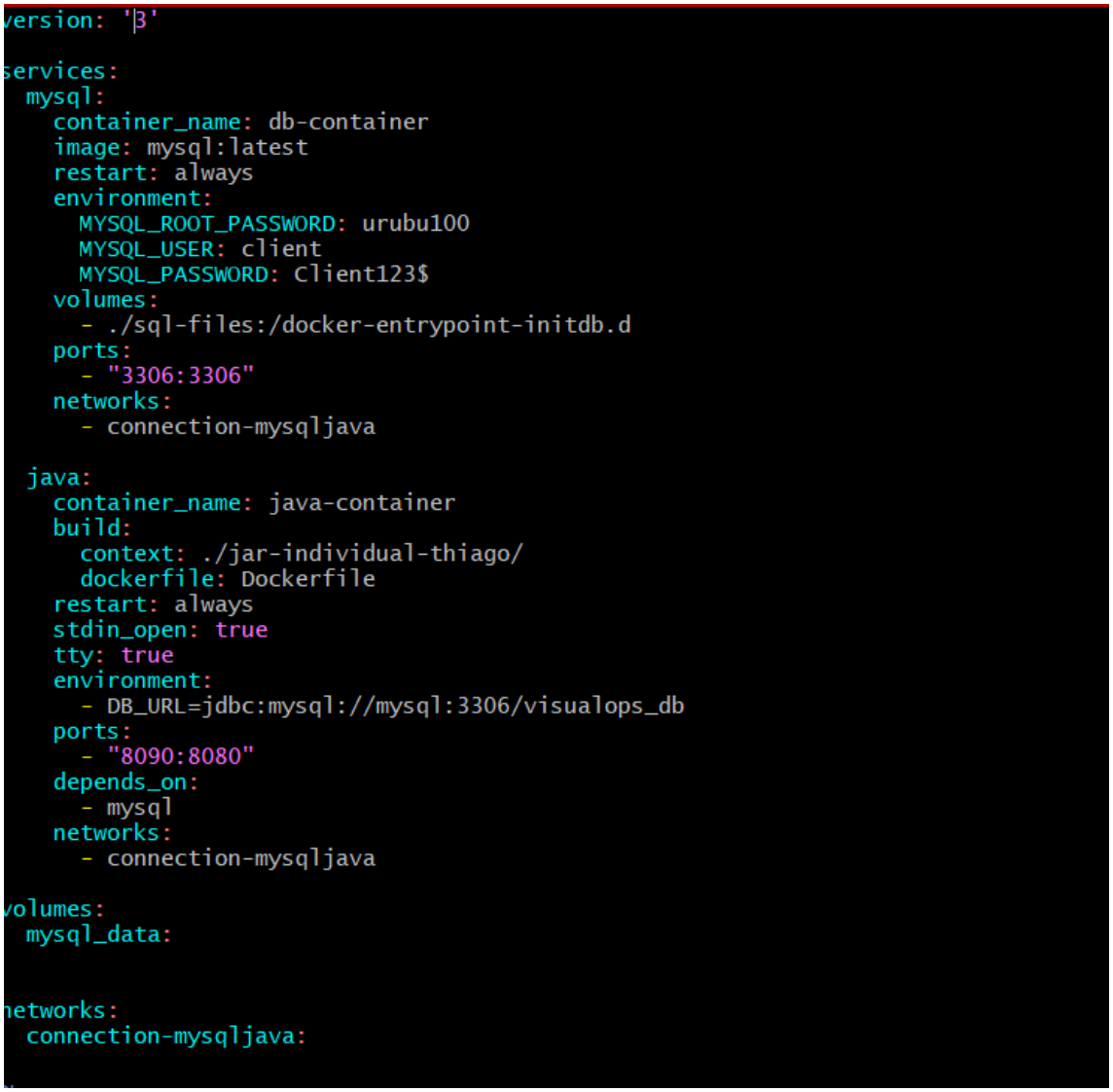
```
FROM maven:3.8.5-openjdk-17 AS builder

WORKDIR /app

COPY . /app

RUN mvn clean install

FROM openjdk:23-ea-17-jdk

COPY --from=builder /app/target/jar-individual-thiago-1.0-SNAPSHOT-jar-with-dependencies.jar /app.jar

CMD ["java", "-jar", "/app.jar"]
```

Agora volte para o seu diretório, mude para o mysql-config e altere o endereço :

```
tigo@ip-172-31-61-112:~/visualops-files$ cat jar-individual-thiago/src/main/java/util/database/MySQLCon
nection.java
package util.database;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class MySQLConnection {
        private static final Logger logger = Logger.getLogger(MySQLConnection.class.getName());

        private static boolean hml = false;

        private static final String CONNECTION_STRING = "jdbc:mysql://mysql:3306/visualops_db";
        private static final String USERNAME = "admin";
        private static final String PASSWORD = "urubu100";

        public static Connection ConBD() throws SQLException {
                try {
                        if (hml){
                                return DriverManager.getConnection(CONNECTION_STRING, USERNAME, PASSWOR
D);
                        } else {
                                return DriverManager.getConnection("jdbc:mysql://mysql:3306/visualops_d
b", "client", "Client123$");
                        }
                } catch (SQLException e) {
                        logger.log(Level.SEVERE, "Erro ao conectar ao banco de dados", e);
                        throw e;
                }
        }
}
tigo@ip-172-31-61-112:~/visualops-files$
```

vim src/main/java/útil/database/MySQLConnection.java

Depois volte para o seu diretório e edite o Docker-compose.yml. vim docker-compose.yml:



```
version: '3'

services:
  mysql:
    container_name: db-container
    image: mysql:latest
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: urubu100
      MYSQL_USER: client
      MYSQL_PASSWORD: Client123$
    volumes:
      - ./sql-files:/docker-entrypoint-initdb.d
    ports:
      - "3306:3306"
    networks:
      - connection-mysqljava

  java:
    container_name: java-container
    build:
      context: ./jar-individual-thiago/
      dockerfile: Dockerfile
    restart: always
    stdin_open: true
    tty: true
    environment:
      - DB_URL=jdbc:mysql://mysql:3306/visualops_db
    ports:
      - "8090:8080"
    depends_on:
      - mysql
    networks:
      - connection-mysqljava

volumes:
  mysql_data:


networks:
  connection-mysqljava:
```

```
version: '3.3'

services:
  mysql:
    container_name: db-container
    image: mysql:latest
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: urubu100
      MYSQL_USER: client
      MYSQL_PASSWORD: Client123$
    volumes:
      - ./sql-files:/docker-entrypoint-initdb.d
    ports:
      - "3306:3306"
    networks:
      - connection-mysqljava

  java:
    container_name: java-container
    build:
      context: ./jar-individual-SEU_NOME/
      dockerfile: Dockerfile
    restart: always
    stdin_open: true
    tty: true
    environment:
```

```
      - DB_URL=jdbc:mysql://mysql:3306/SEU_DATABASE
    ports:
      - "8090:8080"
    depends_on:
      - mysql
    networks:
      - connection-mysqljava

volumes:
  mysql_data:


networks:
  connection-mysqljava:
```

Volte novamente para seu diretório rode o container java:

```
tigo@ip-172-31-61-112:~/visualops-files$ docker-compose run -it java
```

Agora Aprecie!!

Para mais informações ou erros chame no zap.