Java Persistence API (JPA)

API para interação com formas de armazenamento persistente

- os casos mais simples e frequentes são tratados com muito pouco código
- menor necessidade de atualização de código se surgirem novos campos num objeto
- menor dependência do SGBD

https://www.oracle.com/technical-resources/articles/java/jpa.html

JPA e Postgres

0- Obtenha o código inicial (projeto maven)

1- Deve criar uma BD em Postgres, de acordo com:

- dbname: db1
- username: user1
- password: sample1

```
--------------------------------
Sugestão:
--------------------------------
$ sudo su - postgres
postgres@maquina:~$ createuser user1
postgres@maquina:~$ createdb -O user1 db1
postgres@maquina:~$ psql db1
db1=# alter user user1 with password 'sample1';
ALTER ROLE
db1=# \q
postgres@maquina:~$ logout
$
```

```
# para testar o acesso "remoto" à nova BD, do terminal:
```

```
$ psql -h localhost -U user1 db1
# e depois a password
--------------------------------
```

2- Estude as classes:

- Book - dados
  - repare as anotações: Id
- BookRepository - representará a interface para o armazenamento persistente
  - veja os métodos herdados de CrudRepository
    - findAll, save, findById, delete...
  - e métodos locais, com particularidades dos dados e da aplicação local
- StartApplication - exemplo de aplicação que usa dados armazenados em Postgres
  - método run()
    - escrever registos de novos Book
    - consultas à BD

3.a- Compile e execute a aplicação.

mvn compile

3.b- Execute a aplicação.

mvn spring-boot:run

Erros frequentes:

```
ERROR com.zaxxer.hikari.pool.HikariPool - HikariPool-1 - Exception during pool initialization.
org.postgresql.util.PSQLException: FATAL: password authentication failed for user "user1"
```

Veja acima, as instruções para criar um perfil de utilizador, a BD, e a respetiva password... ou ajuste a configuração em src/main/resources/application.properties

---

4- Consulte a BD

```
$ psql -U user1 -h localhost db1
Password for user user1:
psql (12.5 (Ubuntu 12.5-0ubuntu0.20.10.1))
Type "help" for help.
db1=> \dt
       List of relations
 Schema | Name | Type  | Owner
--------+------+-------+-------
 public | book | table | user1
(1 row)
db1=> select * from book ;
 id |  name
----+--------
  1 | Java
  2 | Node
  3 | Python
(3 rows)
db1=> exit
--------------------------------
```

5- O programa continua a fazer reset aos dados no início.
Para não apagar, ir às propriedades, e comentar:
#spring.jpa.hibernate.ddl-auto=create

Na aplicação, em run(), comentar a criação dos livros.

6- Volte a executar

7- Aceda à BD, via psql.
Adicione um livro, ou altere o titulo de um livro...

Volte a executar a aplicação para verificar a leitura desses dados modificados.

Notas:

- Note-se que não somos nós a colocar o driver de postgres na classpath.
  São as ferramentas que usamos. Mas devemos ter consciência do que acontece.
- Ver conteúdo do jar final
- E se quiser separar as configurações do JAR da aplicação: Ver aqui.

Leitura complementar: https://attacomsian.com/blog/spring-data-jpa-h2-database