JPA

Vamos conhecer o JPA, parte integrante do Ecossistema Spring, responsável por criar e manipular o Banco de dados da aplicação.

1. O que é JPA?

JPA (ou *Java Persistence API*) é uma especificação que descreve como deve ser o comportamento dos frameworks de persistência Java que desejarem implementá-la, ou seja, não possui código que possa ser executado.

No Spring, o JPA é uma biblioteca que persiste e recupera objetos que são armazenados em um Bancos de dados. Ele é responsável por fazer todas as instruções SQL sem que precisemos escrever uma única linha em nosso código SQL.

A Biblioteca JPA é uma Interface que possui alguns Métodos assinados, que são os Métodos padrão da Interface (findAll, find ById, Save, deleteById e etc). Como toda e qualquer Interface, para utilizar os Métodos é necessário implementá-los dentro de uma Classe. No contexto Spring, nas Classes das Camadas Controller e Service.

Apesar de ser uma Interface e não possuir nenhum Método Implementado, a especificação possui ainda algumas Classes, Interfaces e Anotações que ajudam o desenvolvedor a criar instruções SQL. Para persistir os dados com JPA, é preciso escolher uma implementação que é quem, de fato, vai fazer todo o trabalho, de acordo com o tipo de Banco de dados que será utilizado (Relacional ou Não Relacional).

1.1 Mapeamento Objeto Relacional

Mapeamento Objeto Relacional é a representação de uma Tabela de um Banco de dados Relacional (MySQL, PostgreSQL, Oracle, SQL Server e etc), através de Classes Java, que dentro do contexto Spring e seguindo o Modelo MVC, são implementadas na Camada Model. Essa técnica de mapeamento também é conhecida como **ORM** ou *Object Relational Mapping*.

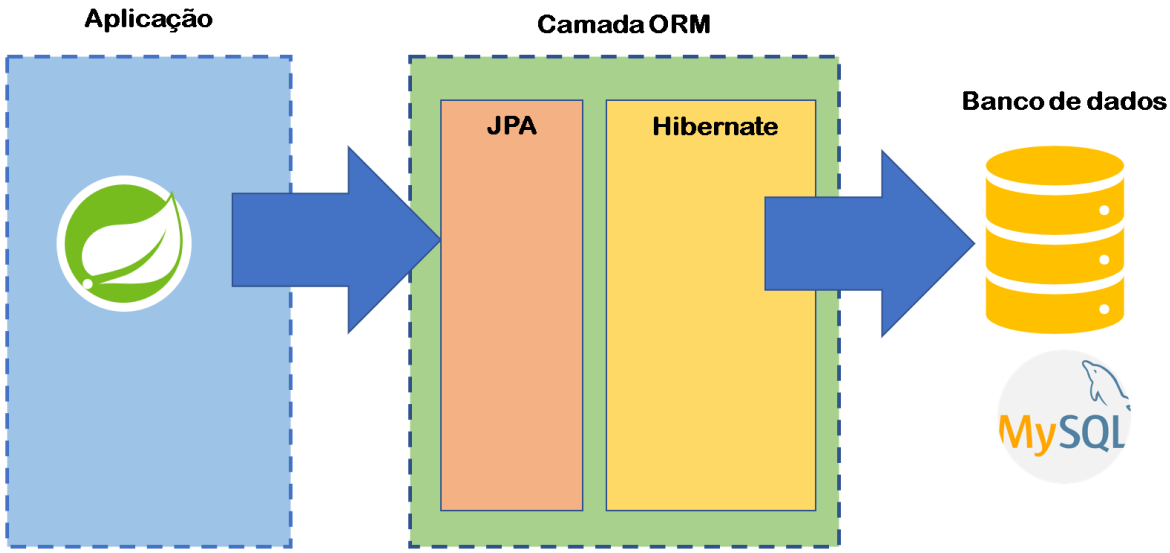
Na prática, o Mapeamento cria a Relação de equivalência abaixo:

Banco de dados		**Linguagem Orientada a Objetos**
Tabela	→	Classe
Coluna	→	Atributo
Registro	→	Objeto

Enquanto que no banco de dados temos Tabelas, Colunas e Registros, em uma linguagem Orientada a Objetos, como o Java, temos o equivalente com Classes, Atributos e Objetos. Além dessa equivalência, o JPA utiliza algumas anotações que adicionarão metadados (Chave Primária, Nome da Tabela, Auto Incremento e etc), às classes e permitirão os frameworks ORM, como o Hibernate, entrarem em ação na geração das tabelas dentro do nosso Banco de dados.

1.2. Hibernate

O **Hibernate** é um framework para o mapeamento objeto-relacional escrito na linguagem Java, cujo objetivo é diminuir a complexidade entre os programas Java, baseado no modelo Orientado a Objetos, que precisam trabalhar com um banco de dados do modelo Relacional (presente na maioria dos SGBD's). Em especial, no desenvolvimento de consultas e atualizações dos dados.



Por quê utilizar JPA?

Escrever SQL para consultas, inserções e atualizações, por mais que não seja complexo, é uma tarefa repetitiva e chata.

A primeira vantagem é que o JPA trás códigos SQL para consultas, inserções e atualizações básicas prontas.

A segunda vantagem é que para criarmos consultas personalizadas podemos utilizar as **Query Methods**, que a partir da combinação de palavras chave, que substituem as instruções SQL, podemos criar consultas de forma rápida.

A terceira vantagem é a simplificação do código-fonte da aplicação. Como muita coisa é automatizada, bem menos código é escrito para as mesmas funções.