**Anotações**

**OneToMany**

**ManyToOne**

Exemplo Funcionario\_Dependente:

OneToMany: Um funcionário pode ter muitos dependentes? Sim

OneToMany: Um dependente pode ter muitos funcionários? Não

ManyToOne: Muitos Funcionários podem ter um dependente? Não

ManyToOne: Muitos Dependentes podem ter um funcionário? Sim

Então na classe Funcionário utiliza o OneToMany com uma lista de dependentes

Então na classe Dependente utiliza o ManyToOne com uma variável funcionário do tipo Funcionario

**JoinColumn()**

Exemplo Funcionario\_Dependente:

Utilizado para acrescentar a chave estrangeira, então deve ser usado no lado que receberá a chave estrangeira no caso o lado do “N”

**Serializable**

Exemplos Funcionario\_Dependente:  
A classe dependente possui uma chave composta baseada no cpf do funcionário “fcpf” e no nome do dependente, então na Classe @Embeddable (dependente\_id) terá as duas chaves primarias

Usado em classes @Embeddable quando se está utilizando chave primária COMPOSTA

**MapsId**

Exemplo Funcionario\_Dependente:

De acordo com o @OneToMany e o @ManyToOne das duas classes, já vai saber quais classes estão acontecendo a conexão. O MapsId automaticamente pega a chave primária da outra classe e preenche a variável onde o nome está correspondente, exemplo:

@MapsId(“funcionário\_cpf”) -> O “funcionário\_cpf” está presente na Classe DependenteId que é uma classe que tem chave completa composta, então o maps automaticamente pega da Funcionario a chave primaria e joga os dados dentro da variável “funcionário\_cpf” da classe DependenteId

**Model**

Contém as entidades do banco que vão ser mapeadas pelo JPA

**Repository**

Contém as interfaces que possuem todas operações básicas de CRUD

**Controller**

Contém as classes que fazem o direcionamento de dados para os EndPoints

**Service**

Contém as classes que realizam as regras de negócio baseado nas interfaces do repositor, exemplo: Validar dados, Calcular valores, Aplicar desconto em vendas.