**Roteiro\_Aula\_JPA\_02**

Alunos: Juliana Vieira; Sérgio Macedo.

RESPONDA:

**I. O que é uma Anotação ou Annotation? Quais as vantagens de usar Annotations no Mapeamento Objeto Relacional?**

São informações sobre o próprio código que é usado para objetos serem usados na compilação, exemplo: informar um objeto relacional.

**II. Sobre as anotações que você conhece:**

**a) Qual delas indica que uma propriedade é Chave Primária?**

@Id

**b) Qual delas indica que o valor é auto-incrementado?**

@GeneratedValue

**c) Qual delas indica que uma propriedade é do tipo Data/Tempo? Que opções a enumeração javax.persistence.TemporalType fornece para indicar a precisão da data? Exemplifique dados temporais que poderiam usar cada dessas opções.**

@Temporal(javax.persistence.TemporalType.DATE)

Podem ser DATE, TIME, TIMESTAMP

**d) Qual delas permite parametrizar a coluna mapeada na propriedade? Dê um exemplo.**

@Column, e informar qual propriedade desejada.

Exemplo: @Column(length = 30)

**e) Qual delas permite parametrizar a tabela mapeada pela classe? Dê um exemplo.**

@Table, e informar qual propriedade desejada.

@Table(name = "veiculo")

**III. Qual o código de uma classe Java anotada para ser mapeada para a tabela abaixo? Para economizar espaço na resposta, não inclua getters e setters da classe, apenas suas propriedades.**

**CREATE TABLE modelo\_de\_veiculo(**

**id\_modelo bigint IDENTITY PRIMARY KEY,**

**descricao VARCHAR(80) NOT NULL );**

import java.io.Serializable;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import org.hibernate.annotations.Columns;

import sun.security.util.BigInt;

@Entity

public class modelo\_de\_veiculo implements Serializable {

@Id

@GeneratedValue

private Long id\_modelo;

@Column(length = 80, nullable = false)

private String descricao;

}

**IV. Escreva a DDL (comando create table) da tabela obtida no final da Fase 3 desse roteiro.**

CREATE TABLE veiculo (

id BIGINT PRIMARY KEY,

placa VARCHAR(20) NOT NULL,

modelo VARCHAR(60) NOT NULL,

cor VARCHAR(30) ,

chassi VARCHAR ,

ano\_de\_fabricacao DATE,

quilometragem INTEGER,

)

**RESPONDA:**

**I. Que anotação é utilizada para representar o mapeamento Muitos-Para-Um?**

@ManyToOne

**II. Qual seria o relacionamento contrário ao Muitos-Para-Um? Na prática, o que ele significa? Que tipo de propriedade pode ser mapeada com ele?**

Um para muitos (OneToMany). Que o registro de uma entidade está relacionado com vários registros de outra entidade. Vão ser listas.

**III. Que anotação é utilizada para representar o mapeamento Muitos-Para-Muitos?**

@ManyTo Many

**IV. Que outra forma existe de mapear a relação Muitos-Para-Muitos?**

Criando duas classes mapeadas com @OneToMany e associá-las a uma classe de entidade fraca. (Pode-se criar uma terceira classe para fazer o mapeamento das tabelas)

**V. Pesquise e descubra sobre os tipos de carregamento de entidades relacionadas:**

**FetchType.Lazy e FetchType.Eager. O que cada um deles significa? Onde eles podem ser configurados?**

No FetchType.Lazy essa opção quando descrita na propriedade do mapeamento permite que quando um objeto for acessado ele não traga instantaneamente todas as dependências, somente quando for preciso, economizando memória.

No FetchType.Eage essa opção quando descrita na propriedade do mapeamento traz todas as dependências instantaneamente de um objeto quando ele é acessado, o que ocupa um grande espaço de memória.

Eles podem ser configurados nas propriedades dos relacionamentos, exemplo, @ManyToOne(fetch = FetchType.LAZY).

**VI. Em que casos seria interessante um relacionamento Um-Para-Um? Pesquisa e descubra qual anotação pode ser utilizada para denotar esse relacionamento.**

Quando o tipo de relacionamento for um 1:1, exemplo, quando um cliente pode ter apenas um usuário. @OneToOne.