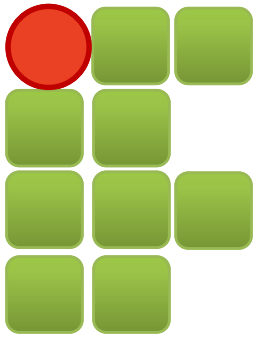
INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
BAHIA

Banco de Dados

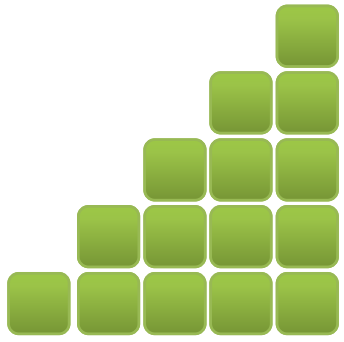
Prof. Eliomar Campos



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
BAHIA

Herança em BD Relacionais

Tipos de mapeamentos possíveis para herança



Introdução

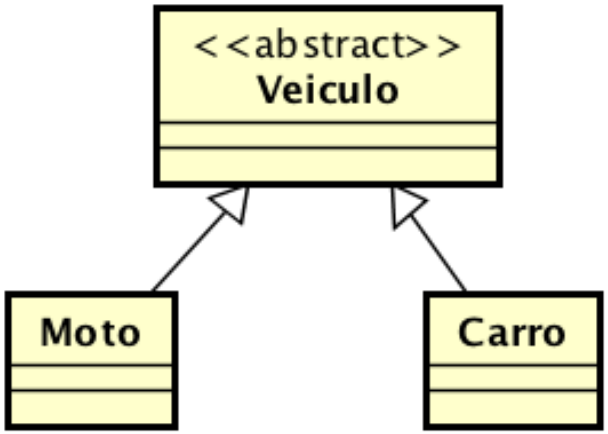
- Um dos grandes desafios em desenvolver um software baseado em orientação a objetos é abstrair os dados de sua base para objetos, afim de poder melhor manipula-los;

- Hoje os frameworks ORM (Object Relational Mapper), ou mapeador de objeto relacional, disponíveis facilitam bastante esta abstração;

- O problema mesmo é quando começamos a modelar o sistema a partir de objetos e principalmente quando damos hierarquia a eles utilizando heranças, e depois como representar essas hierarquias no banco de dados relacional?

Introdução

- Por exemplo:



- Qual a melhor maneira de representar estas hierarquias no banco de dados relacional, uma tabela para cada tipo? mesmo Veiculo sendo uma classe abstrata?

Soluções

- Existem várias maneiras de mapear a relação de herança no banco;
- A melhor estratégia vai depender da situação:
  - Tipos de consultas que serão efetuadas contra os dados;
  - Quantidade de campos comuns Vs campos específicos;
  - Tamanho da hierarquia;
  - Quantidade de dados, etc.

Solução 1

- **Tabela por entidade:**
- Cada tabela conterá não só os dados da classe filha como os dados da classe pai. Isso permite que consultas "independentes" entre carro e motos sejam mais simples e rápidas.

Solução 2

- **Tabela única + coluna discriminadora:**

- Uma única tabela conterá todos os dados de todas as entidades;
- Para diferenciar entre carros e motos usamos uma coluna discriminadora;
- Isso permite que consultas "genéricas" (i.e., sobre todos os veículos) sejam mais simples e rápidas;
- É especialmente apropriada quando existem poucos atributos específicos de carro e moto (evitando assim uma grande quantidade de valores nulos).

Solução 3

- **Tabela principal + tabela filhas com FK para a tabela principal:**

- Esse modelo é apropriado quando normalização for essencial e as tabelas filhas tiverem muitas colunas;
- Além disso essa estratégia é especialmente interessante para trabalhar com herança múltipla (uma entidade pode ser de vários tipos na hierarquia simultaneamente);
- A fraqueza dessa modelagem é que consultas requerem *join*, o que torna o código complexo e traz problemas de *performance* conforme a quantidade de dados e a profundidade da hierarquia vai crescendo.

Solução 4

- **Estratégias mistas:**

- Nada impede que você modele parte da sua hierarquia conforme uma estratégia e parte conforme outra.
- Por exemplo, você pode ter uma tabela principal veículo com os dados comuns à todos os veículos (velocidade, tamanho, peso, etc.), porém, ter tabelas únicas para cada tipo de veículo (veículos terrestres, aquáticos, voadores, etc.).

Conclusão

- Todas as estratégias possuem suas vantagens e suas desvantagens; representações OO e relacionais são suficientemente diferentes para que existam vários problemas de mapeamento;
- Portanto, o correto é tentar evitar maiores problemas se adequando ao máximo a cada necessidade.