MER – Modelo Entidade-Relacional

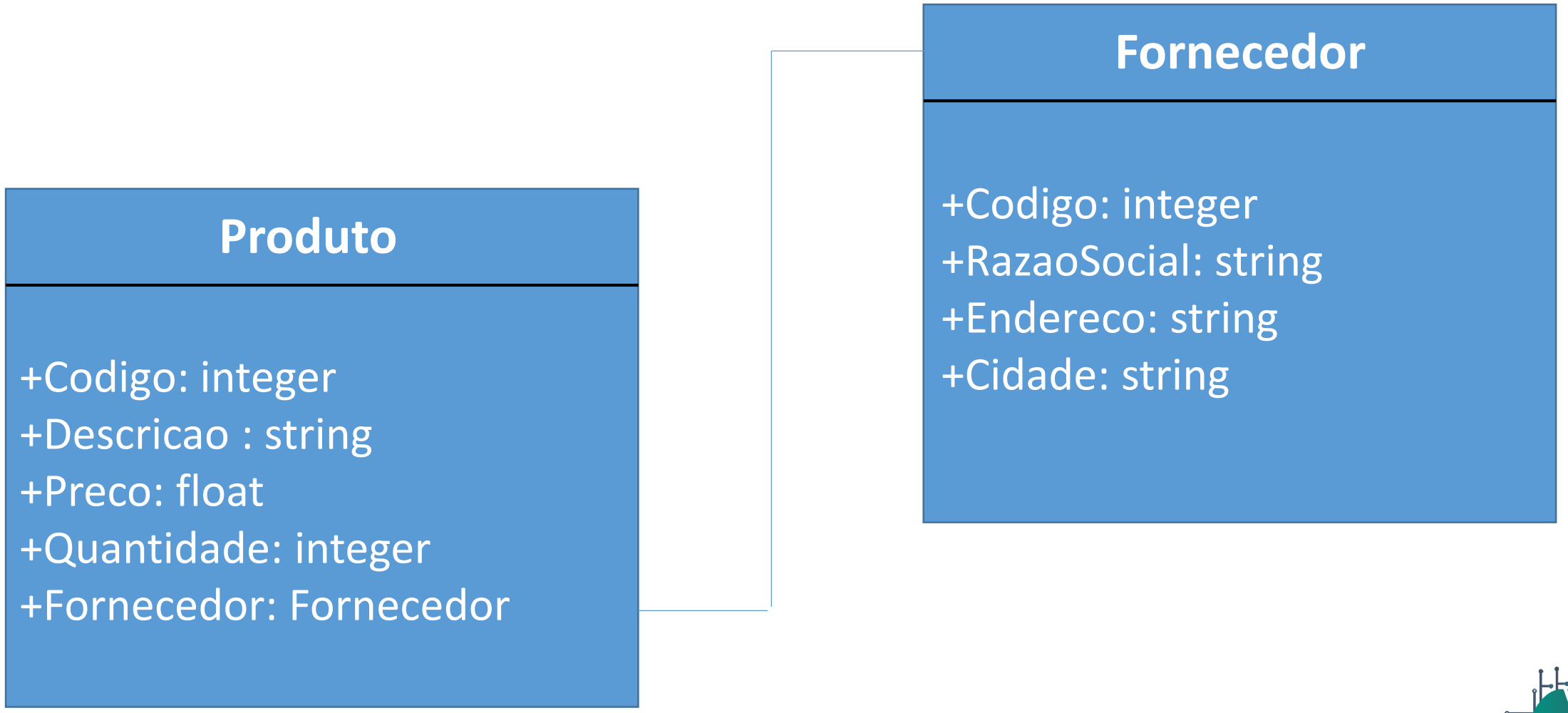
Como os objetos podem se comunicar entre si

TECH
soluções em ti

Associação

É a relação mais comum entre dois objetos, de modo que um possui uma referência à posição da memória onde o outro se encontra, podendo visualizar seus atributos ou mesmo acionar um dos seus métodos.

A forma mais comum de se implementar uma associação é ter um objeto como atributo de outro. No exemplo a seguir, criaremos duas classes: Produto e Fornecedor. Um dos atributos do produto é o fornecedor, o qual estará associado a classe Fornecedor.

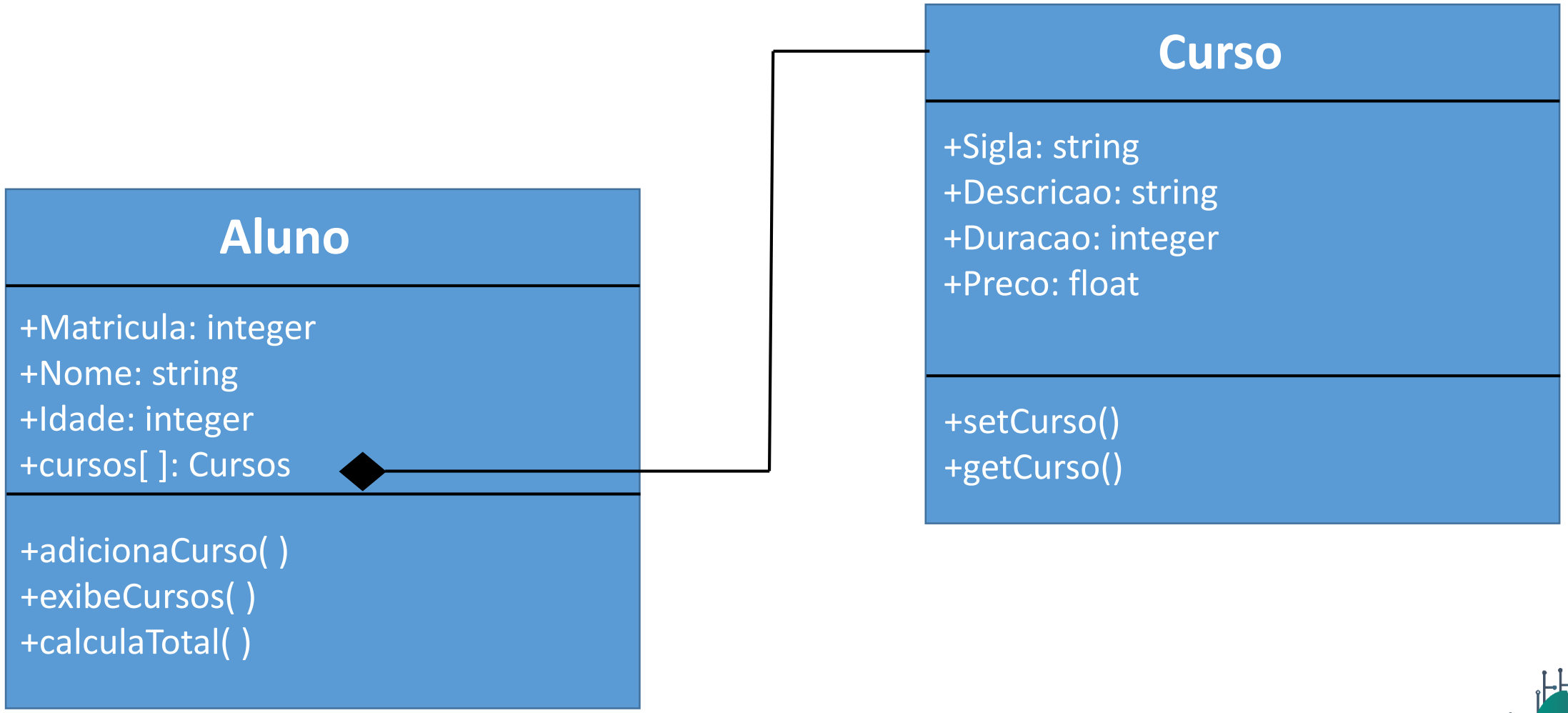


TECH
soluções em ti

Agregação

É o tipo de relação entre objetos conhecida como todo/parte. Na agregação, um objeto agrega outro objeto, ou seja, torna um objeto externo parte de si mesmo pela utilização de um dos seus métodos. Assim, o objeto-pai poderá utilizar funcionalidades do objeto agregado.

Nesta relação, um objeto poderá agregar uma ou muitas instâncias de um outro objeto. Para agregar muitas instâncias, a forma mais simples é utilizando arrays. No exemplo a seguir, criaremos duas classes: Aluno e Curso. Um aluno poderá escolher inúmeros cursos (instâncias da classe Curso). Para agregar objetos do tipo curso ao Aluno, criaremos o método adicionaCurso() na classe Aluno, que conta também com o método exibeCursos(), no qual chama o método getCurso() de cada um dos cursos, e o método calculaCursos(), que soma o valor de cada um dos cursos de um aluno.



Talvez você já tenha percebido um possível problema neste tipo de abordagem. O que aconteceria se adicionássemos objetos que não são da classe Curso ao aluno?

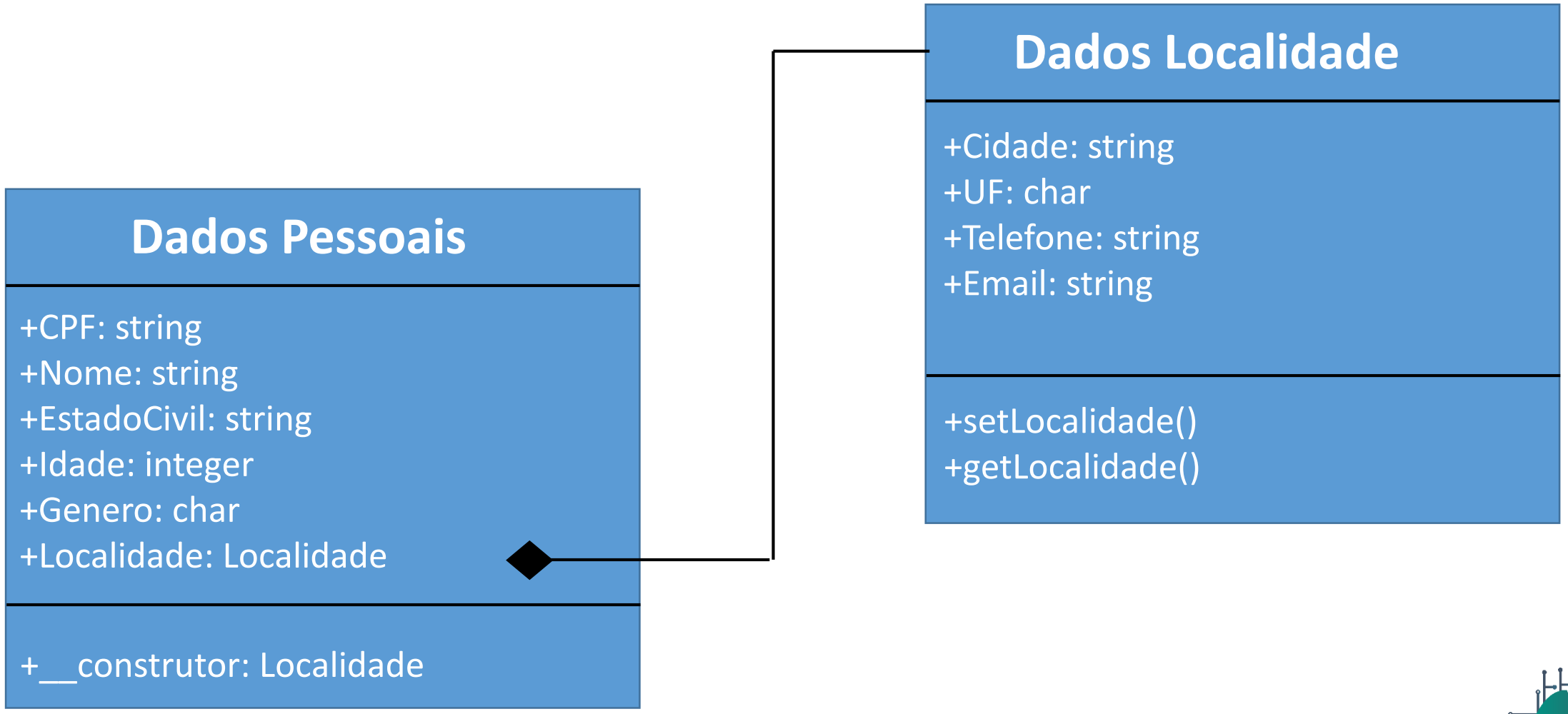
Se chamássemos métodos como calculaCursos() e exibeCursos, por exemplo, teríamos problemas, uma vez que eles confiam na existência do atributo Preco e do método getCurso() da classe Curso.

O PHP Moderno introduz o conceito de TypeHinting, ou seja, “sugestão de tipo”. Nesse exemplo da classe Aluno, o método adicionaCurso() recebe um curso chamado $curso, e na frente do parâmetro indicamos a classe à qual ele dever pertencer.

Composição

Também é uma relação que demonstra todo/parte. A diferença é que, na composição, o objeto-pai ou “todo” é responsável pela criação e destruição de suas partes. O objeto-pai realmente “possui” a(s) instância(s) de suas partes. Diferentemente da agregação, na qual as instâncias do “todo” e das “partes” são independentes.

Na agregação, ao destruirmos o objeto “todo”, as “partes” permanecem na memória por terem sido criadas fora do escopo da classe “todo”. Já na composição, quando o objeto “todo” é destruído, suas “partes” também são, justamente por terem sido criadas pelo objeto “todo”.



TECH
soluções em ti

PDO::PHP Data Object

Em razão da crescente adoção do PHP, surgiu a necessidade de unificar o acesso às diferentes extensões de bancos de dados presentes no PHP. Assim surgiu a PDO, cujo objetivo é prover uma API limpa e consistente, unificando a maioria das características presentes nas extensões de acesso a banco de dados.

A PDO não é uma biblioteca completa para abstração do acesso à base de dados, uma vez que ela não faz a leitura e tradução das instruções SQL, adaptando-as aos mais diversos drivers de bancos existentes. Ela simplesmente unifica a chamada de métodos, delegando-os para as suas extensões correspondentes e faz uso do que há mais de recente no que diz respeito a OO no PHP Moderno.

Principais métodos da classe PDO

Construtor da classe: permite instancia uma nova conexão com banco de dados. Sintaxe: new PDO($dsn, $user, $password, $options)

Prepare() -> utilizado para preparar um statement(insert, update, delete) que será executado

Execute() -> executa uma instrução SQL foi preparada pelo método prepare()

Query() -> utilizado para realizar consultas de um determinado statement(select)

Fetch() -> retorna os dados que foram selecionados por uma chave-primária

fetchAll() -> retorna os dados que foram selecionados

bindValue() -> realiza uma busca por um valor, dentro da sintaxe SQL

bindParam() -> realiza uma busca por um valor, dentro da sintaxe SQL

rowCount() -> retorna a quantidade de registros encontrados

TECH
soluções em ti

PDO::FETCH_ASSOC -> retorna um array indexado pelo nome da coluna

PDO::FETCH_NUM -> retorna um array indexado pela posição numérica da coluna

PDO::FETCH_BOTH -> retorna um array indexado pelo nome da coluna e pela posição numérica da mesma

PDO::FETCH_OBJ -> retorna um objeto anônimo(stdClass), de modo que cada coluna é acessada como um atributo do objeto

Podemos utilizar o laço que percorre um array para retornar todos os registros que forem encontrados em uma entidade (tabela)