**Relacionando Produtos com Categorias**

O nosso sistema precisa evoluir. Nesse momento, já temos um sistema que faz todo o controle de produtos. Mas agora precisamos fazer com que cada produto tenha uma categoria associada.

Nosso primeiro passo será criar mais uma tabela no banco de dados, afinal precisamos salvá-las em algum lugar. No PhpMyAdmin:

create table categorias (id integer auto\_increment primary key, nome varchar(255));

Vamos também já popular a tabela e inserir algumas categorias:

insert into categorias (nome) values ("esporte"), ("escolar"), ("mobilidade");

Agora, precisamos exibir as categorias disponíveis no formulário de adição de produtos. Faremos isso com aquelas "bolinhas", que permitem com que o usuário selecione uma dentre as várias opções. Essas bolinhas são chamadas de radio button. Vamos adicionar essa linha no formulário:

<tr>

<td>Categoria</td>

<td>

<input type="radio" name="categoria">Esporte</br>

<input type="radio" name="categoria">Escolar</br>

<input type="radio" name="categoria">Mobilidade</br>

</td>

</tr>

A lista de categorias já aparece direito no formulário de produtos. Mas precisamos ainda configurar o valor de cada radio, afinal é o valor que é postado para o PHP. É bem comum fazermos uso dos IDs. Cada categoria tem seu ID, então vamos colocar isso no formulário, renomeando inclusive o nome dos radios para deixar isso claro:

<tr>

<td>Categoria</td>

<td>

<input type="radio" name="categoria\_id" value="1">Esporte</br>

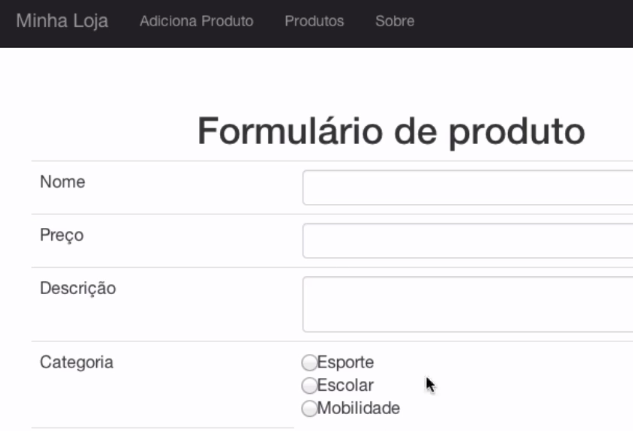
<input type="radio" name="categoria\_id" value="2">Escolar</br>

<input type="radio" name="categoria\_id" value="3">Mobilidade</br>

</td>

</tr>

Apesar de estar funcionando, o problema é que as categorias estão fixas em nosso HTML. Se alguém inserir uma nova categoria no banco de dados, ela não será refletida automaticamente na listagem. Precisamos fazer com que isso seja dinâmico, ou seja, buscar as categorias do banco de dados.



Já sabemos que precisamos criar um novo arquivo para lidar com o acesso aos dados da Categoria. Então, no arquivo banco-categoria.php, fazemos:

<?php

function listaCategorias($conexao) {

$categorias = array();

$query = "select \* from categorias";

$resultado = mysqli\_query($conexao, $query);

while($categoria = mysqli\_fetch\_assoc($resultado)) {

array\_push($categorias, $categoria);

}

return $categorias;

}

Com esse arquivo pronto, podemos usá-lo agora no produto-formulario.php:

<?php

include("cabecalho.php");

include("conecta.php");

include("banco-categoria.php");

$categorias = listaCategorias($conexao);

?>

... codigo continua aqui ...

<tr>

<td>Categoria</td>

<td>

<?php foreach($categorias as $categoria) : ?>

<input type="radio" name="categoria\_id" value="<?=$categoria['id']?>"><?=$categoria['nome']?></br>

<?php endforeach ?>

</td>

</tr>

Se agora voltarmos para o formulário, percebemos que eles vem do banco de dados. Para testar, adicione uma quarta categoria "guloseimas". Veja que ela aparecerá no formulário.

Agora precisamos salvar a categoria que é postada! Vamos começar criando a coluna na tabela produtos. Criaremos lá um campo inteiro que guardará o ID da categoria. É assim que fazemos: sempre guardamos os IDs das outras tabelas. Lembre-se de fazer o curso de SQL se esse conceito for novo pra você.

alter table produtos add column categoria\_id integer

Agora todos os produtos tem categoria NULL. Vamos dar um UPDATE para setar um valor padrão:

update produtos set categoria\_id = 3;

Podemos atualizar as categorias de maneira mais adequada. Mas aqui, para não perdermos tempo, vamos passar reto. Vamos agora no nosso código, fazer com que ele capture o ID da categoria:

$nome = $\_POST['nome'];

$preco = $\_POST['preco'];

$descricao = $\_POST['descricao'];

$categoria\_id = $\_POST['categoria\_id'];

if(insere\_produto($conexao,$nome,$preco,$descricao,$categoria\_id)) {

// ...

}

No banco-produto.php, também recebemos a categoria e modificamos a query:

function insereProduto($nome, $preco, $descricao, $categoria\_id) {

$query = "insert into produtos (nome, preco, descricao, categoria\_id) values ('{$nome}', {$preco}, '{$descricao}', {$categoria\_id})";

$resultadoDaInsercao = mysqli\_query($conexao, $query);

}

Se cadastrarmos um produto agora, conseguimos selecionar uma categoria, e a mesma é salva direito no banco de dados. Podemos ver isso pelo PhpMyAdmin.

Próximo passo agora é exibir a categoria na listagem. Vamos adicionar uma coluna a mais na listagem:

<tr>

<td><?= $produto['nome'] ?></td>

<td><?= $produto['preco'] ?></td>

<td><?= substr($produto['descricao'],0,40) ?></td>

<td><?= $produto['categoria\_id'] ?></td>

<td>

<a href="remove-produto.php?id=<?= $produto['id']?>" class="text-danger">remover</a>

</td>

</tr>

Isso funciona, mas exibe um número! Não é útil para o usuário! Vamos exibir o nome. Para isso, precisamos mudar o SELECT que traz o produto, e fazer com que ele também traga o nome da categoria. Aqui faremos uma SQL mais avançada:

<?php

function listaProdutos($conexao) {

$produtos = array();

$resultado = mysqli\_query($conexao, "select p.\*, c.nome as categoria\_nome from produtos as p join categorias as c on p.categoria\_id = c.id");

while($produto = mysqli\_fetch\_assoc($resultado)) {

array\_push($produtos, $produto);

}

return $produtos;

}

?>

Agora na listagem, exibimos o campo categoria\_nome, que definimos na consulta SQL:

<tr>

<td><?= $produto['nome'] ?></td>

<td><?= $produto['preco'] ?></td>

<td><?= $produto['descricao'] ?></td>

<td><?= $produto['categoria\_nome'] ?></td>

<td>

<a href="remove-produto.php?id=<?= $produto['id']?>" class="text-danger">remover</a>

</td>

</tr>

Agora sim, a categoria aparece corretamente na nossa listagem!