

IF no SQL + Lógica dos Logins

 por Rafael SP



IF no SQL e Lógica de Sistemas de Login

1. O que é o IF no SQL?

O **IF** no SQL é utilizado para realizar uma comparação e, com base no resultado dessa comparação, decidir qual valor será retornado. Em outras palavras, o **IF** funciona de maneira condicional dentro das consultas SQL. A sintaxe básica do **IF** é:

Sintaxe:

```
SELECT IF(condição, valor_se_verdadeiro, valor_se_falso);
```

- **condição:** A condição que será avaliada (ex: `campo = 1`).
- **valor_se_verdadeiro:** O valor que será retornado se a condição for verdadeira.
- **valor_se_falso:** O valor que será retornado se a condição for falsa.

Exemplo:

Imagine que temos uma tabela de usuários e queremos verificar se o status de um usuário é "ativo". Podemos usar o **IF** para retornar um valor condicionalmente.

```
SELECT nome, status,  
       IF(status = 'ativo', 'Usuário ativo', 'Usuário inativo') AS status_usuario  
FROM usuarios;
```

Neste exemplo, estamos verificando o campo `status`. Se for "ativo", o resultado será "Usuário ativo". Caso contrário, será "Usuário inativo".

Como Funciona a Lógica de um Sistema de Login?

Em um sistema de login, a lógica principal é verificar as credenciais do usuário (normalmente **nome de usuário** e **senha**) e, com base nisso, decidir se o usuário tem permissão para acessar o sistema. O processo geralmente envolve:

1. **Captura de dados do formulário de login.**
2. **Verificação das credenciais no banco de dados.**
3. **Criação de uma sessão ou token de acesso.**
4. **Retorno de uma mensagem para o usuário.**

Agora, vamos entender como isso é feito com PHP e MySQL.

1. Criação da Tabela de Usuários (MySQL)

Primeiro, criamos uma tabela simples de usuários no MySQL:

```
CREATE TABLE usuarios (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  nome_usuario VARCHAR(255) NOT NULL,  
  senha VARCHAR(255) NOT NULL,  
  email VARCHAR(255) NOT NULL,  
  status ENUM('ativo', 'inativo') NOT NULL DEFAULT 'ativo'  
);
```

Aqui, temos:

- **id**: Identificador único do usuário.
- **nome_usuario**: Nome de usuário único.
- **senha**: A senha do usuário (deve ser armazenada de forma segura, como veremos).
- **email**: E-mail do usuário.
- **status**: Status do usuário, que pode ser "ativo" ou "inativo".

2. Formulário de Login (HTML)

Aqui temos um exemplo simples de formulário de login em HTML:

```
<form action="login.php" method="POST">
  <label for="nome_usuario">Nome de Usuário:</label>
  <input type="text" name="nome_usuario" id="nome_usuario" required>
  <br>
  <label for="senha">Senha:</label>
  <input type="password" name="senha" id="senha" required>
  <br>
  <button type="submit">Entrar</button>
</form>
```

Este formulário envia os dados para o script PHP `login.php`, onde faremos a verificação.

3. Verificação das Credenciais (PHP)

Agora, vamos criar a lógica do PHP para verificar o nome de usuário e senha fornecidos no formulário.

```
<?php
// login.php

session_start(); // Inicia a sessão, para armazenar a autenticação

// Verifique se o formulário foi enviado
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Conectar ao banco de dados MySQL
    $conn = new mysqli('localhost', 'root', '', 'nome_do_banco_de_dados');

    // Verifique a conexão
    if ($conn->connect_error) {
        die("Conexão falhou: " . $conn->connect_error);
    }

    // Escape de entradas para prevenir SQL Injection
    $nome_usuario = $conn->real_escape_string($_POST['nome_usuario']);
    $senha = $conn->real_escape_string($_POST['senha']);

    // Query SQL para verificar as credenciais
    $sql = "SELECT id, nome_usuario, senha, status FROM usuarios WHERE nome_usuario = '$nome_usuario'";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        // A senha fornecida no formulário
        $user = $result->fetch_assoc();

        // Verificando se a senha está correta (comparamos a senha criptografada)
        if (password_verify($senha, $user['senha'])) {
            // Se o status for ativo, cria uma sessão e direciona para a área interna
            if ($user['status'] == 'ativo') {
                $_SESSION['usuario_id'] = $user['id'];
                $_SESSION['usuario_nome'] = $user['nome_usuario'];
                header("Location: area_restrita.php");
                exit;
            } else {
                echo "Seu usuário está inativo.";
            }
        } else {
            echo "Nome de usuário ou senha inválidos.";
        }
    } else {
        echo "Nome de usuário não encontrado.";
    }
}

$conn->close(); // Fecha a conexão
?>
```

Neste código:

- Conectamos ao banco de dados MySQL.
- Pegamos os dados do formulário `POST` (nome de usuário e senha).
- Realizamos uma consulta SQL para procurar o usuário no banco de dados.
- Se o nome de usuário existir, verificamos se a senha fornecida corresponde à senha armazenada no banco de dados (usamos `password_verify()` para comparar as senhas).
- Se o login for bem-sucedido e o status do usuário for "ativo", iniciamos uma sessão e redirecionamos o usuário para a página da área restrita.
- Caso contrário, exibimos mensagens de erro.

4. Armazenando Senhas de Forma Segura

No exemplo acima, usamos `password_verify()` para comparar as senhas. No banco de dados, você deve armazenar a senha **de forma segura** utilizando funções como `password_hash()`:

```
// Ao registrar um novo usuário
$senha_hash = password_hash($senha, PASSWORD_DEFAULT);
$sql = "INSERT INTO usuarios (nome_usuario, senha, email, status) VALUES ('$nome_usuario', '$senha_hash', '$email', 'ativo')";
```

Assim, a senha nunca será armazenada em texto simples no banco de dados, o que é crucial para a segurança.

5. Redirecionamento após o Login

No script `login.php`, se o login for bem-sucedido, usamos `header("Location: area_restrita.php");` para redirecionar o usuário para a área restrita.

Aqui, podemos ter uma página chamada `area_restrita.php`:

```
<?php
session_start();

// Verifique se o usuário está logado
if (!isset($_SESSION['usuario_id'])) {
    header("Location: login.php");
    exit;
}

// Se o usuário está logado, exiba a página restrita
echo "Bem-vindo à área restrita, " . $_SESSION['usuario_nome'];
?>
```

Conclusão

Agora, vimos como funciona o **IF no SQL** para condicionalmente retornar valores em uma consulta. Também exploramos como criar um **sistema de login simples com PHP e MySQL**, onde:

- Criamos uma tabela de usuários no banco de dados.
- Criamos um formulário de login.
- Verificamos as credenciais no PHP, usando SQL e funções de segurança como `password_hash` e `password_verify`.

Esse é um exemplo básico, mas há muitos aprimoramentos possíveis, como adicionar medidas de segurança mais avançadas, como **limitar tentativas de login**, **autenticação de dois fatores**, etc.

Se precisar de mais algum detalhe ou tiver dúvidas, é só me chamar!