LIKE é um comando SQL usado na cláusula WHERE para procurar um padrão específico em uma coluna. Em outras palavras, é uma maneira de fazer uma pesquisa em uma tabela, da mesma maneira que você pode usar um filtro de pesquisa em uma planilha do Excel, ou utilizar o método contains() do Java.

Você pode estar se perguntando: "Por que eu deveria aprender sobre LIKE se eu simplesmente posso procurar diretamente a informação que quero na minha tabela?". Bom, imagine que você tem uma tabela com milhares de registros, e você precisa encontrar todos os nomes que começam com a letra 'A'. Fazer isso manualmente levaria muito tempo, não é? É aqui que o comando LIKE é realmente útil!

Com o comando LIKE, você pode simplificar essa tarefa fazendo uma consulta que retorna apenas os registros que correspondem ao padrão que você define.

A sintaxe básica para usar o comando LIKE é a seguinte:

**SELECT** column1, column2, ...

**FROM** table\_name

**WHERE** **column** **LIKE** **pattern**;

COPIAR CÓDIGO

O 'pattern' é o padrão que você está procurando. Para definir esse padrão, você usa os caracteres curinga % e \_. O % substitui zero ou mais caracteres, enquanto \_ substitui um único caractere.

Por exemplo, para encontrar todos os nomes em uma tabela de clientes que começam com 'A', você usaria:

**SELECT** nome

**FROM** clientes

**WHERE** nome **LIKE** 'A%';

COPIAR CÓDIGO

Isso retornará todos os registros na coluna 'nome' que começam com 'A'.

Se você quiser todos os nomes com 'A' em qualquer lugar do nome, você usaria:

**SELECT** nome

**FROM** clientes

**WHERE** nome **LIKE** '%A%';

COPIAR CÓDIGO

Mas onde se encaixa o comando ILIKE que utilizamos em aula? O comando LIKE é sensível a maiúsculas e minúsculas em alguns bancos de dados. Portanto, se você quiser uma busca que não leve isso em consideração, como se fosse o IgnoreCase em Java, você deve utilizar o ILIKE.

Novamente, é importante que a gente entenda o uso dos dois comandos para conseguir compará-los e escolher o que melhor se adequa à nossa realidade de pessoas desenvolvedoras.

Agora vamos para a questão  
  
Você está trabalhando como desenvolvedor(a) em um projeto de e-commerce de livros chamado Buscante. Sua tarefa é implementar uma funcionalidade que permite às pessoas usuárias buscar livros através do título usando JPQL. No entanto, você precisa garantir que a busca seja **case-insensitive**, ou seja, não diferencia letras maiúsculas de minúsculas.

SELECT livro FROM Livro livro WHERE LOWER(livro.titulo) = LOWER(:titulo)

Esta consulta vai transformar tanto o título do livro quanto a string de busca em letras minúsculas antes de fazer a comparação, tornando-a case-insensitive. Poderíamos também utilizar o ILIKE para fazer essa busca.