05

Habilitando o cache

Transcrição

[00:00] Continuando a ideia que conversamos no vídeo anterior. Quando o usuário acessar a nossa página *home*, toda vez que para abrir essa página para ser apresentado esse conteúdo para ele, nossa aplicação tem que ir no banco de dados e buscar os pedidos em ordem decrescente por data de entrega e exibir na página. Isso está ok.

[00:24] Mas se tivermos muitos usuários, vamos ter muitas consultas ao banco de dados e é muito provável que um preciosismo aqui de estarmos apresentando os últimos pedidos entregues, não seja uma necessidade e, de certa forma, é como se fosse um requisito que nos força a estarmos sempre no banco de dados.

[00:51] Podemos pensar em outra ideia em que a cada 10 segundos as consultas são realmente feitas no banco de dados. Por que isso? Porque se a estivermos em um cenário onde temos 100 usuários por segundo, eu vou ter 100 consultas ao banco de dados por segundo; porque toda vez em que abrimos essa página, temos uma consulta ao banco de dados.

[01:20] Para ficar claro isso, eu posso até habilitar, no "application.properties", no menu à esquerda, eu vou colocar duas configurações aqui do Hibernate: `spring.jpa.properties.hibernate.format_sql=true` e `spring.jpa.properties.hibernate.show_sql=tue` , que apresentam o SQL que ele está executando no banco de dados.

[01:35] O que é isso? Eu vou limpar aqui o *log*. O *log* está limpo, vou para a página *home* e atualizar essa página. Ela parece que não mudou nada, mas a

requisição foi no back-end.

[01:50] E olhe isso, na aba Console da nossa IDE, isso é a impressão do "select". Então estamos fazendo um *select* de pedido, por status, ordenando por data de entrega, com determinado limite. Essa é a implementação dessa query para o MariaDB.

[02:09] Só que os pedidos não mudaram desde a última consulta para essa. Se eu fizer uma nova consulta aqui, se eu abrir essa página de novo, veja que ele faz o *select* de novo. E assim ele vai fazer o tempo inteiro.

[02:22] Então o que eu estou falando é exatamente isso: se eu tiver 100 usuários, eu vou ter 100 consultas no banco de dados. Pode ser que seu banco de dados uma hora não aguente e vai ser de maneira desnecessária, porque o conteúdo não está mudando. Então podemos combinar de consultar o banco de dados só quando o conteúdo mudar, por exemplo.

[02:41] E fazemos isso com o cache. O cache é isso. Você vai fazer uma determinada consulta e você diz para o Spring: "cacheie esse conteúdo", ou seja, "guarde esse conteúdo por tanto tempo ou até ele mudar".

[02:56] E de forma que quando eu chamar de novo uma consulta ao banco de dados - por exemplo: `findByStatus` - ao invés de eu ir no banco, eu vou no cache, que na versão mais simples que temos do Spring ele fica em memória.

[03:13] Então eu vou ter, por exemplo, 10 pedidos em memória, na memória RAM mesmo. E todos os usuários, quando forem acessar, ao invés de irem no banco de dados, eles vão acessar os pedidos que estão salvos em memória. Isso é muito mais rápido e vai gerar muito menos requisições ao banco de dados. Vai fazer nossa aplicação performar e entregar muito melhor. cache é uma coisa muito importante.

[03:44] Vamos habilitar cache aqui na nossa aplicação com o Spring Boot. Eu vou na documentação, para sempre direcionar vocês, para vocês poderem se

aprofundar nesse conteúdo depois.

[03:55] Aqui no menu do site do Spring Framework, em "Projects > Spring Framework", eu vou em "Learn", nessa versão 5.2.8. Aqui ele está quebrado, porque aqui ele tem muito mais assuntos a tratar e eles quebraram em várias documentações diferentes. E essa penúltima, "Integration", fala sobre *Caching*.

[04:14] Nós clicamos na "Integration", vamos no final, "Cache Abstraction" e ele vai falar sobre cache, como funciona e tudo mais. Vai ter toda uma explicação sobre isso e alguns exemplos, como esse `@Cacheable("books")`.

[04:30] Se você quiser cachear, por exemplo, uma busca de livros, você usa essa anotação, `@Cacheable` e você dá um nome para ela - que no caso ele deu "books". Então todas as vezes temos um determinado nome.

[04:44] Além disso, temos que habilitar o cache na nossa aplicação. Eu não vou achar aqui, se eu ficar rolando muito tempo vou perder tempo. Então vou voltar para aquele exemplo que copiamos aqui.

[04:56] Eu vou pegar essa anotação, `@Cacheable("books")`, e vou colar nesse `findByStatus`, de forma que essa consulta seja cacheada.

[05:25] Nós vamos no `@Cacheable` aqui no "PedidoRepository.java". Só que, o que acontece? Não é porque você o anotou com `@Cache` que o cache está funcionando. Para nós realmente conseguirmos fazer o cache funcionar, precisamos vir no "MudiApplication.java" e colocarmos esse `@EnableCaching`. Aí o cache realmente é inicializado na hora em que a nossa aplicação sobe. Ele está falando algumas coisas aqui com relação a cache, isso não é erro.

[05:54] E o que acontece? Vamos acompanhar o log de novo, porque eu vou abrir a página *home* mais uma vez. E aconteceu o que é esperado, ele vai no banco de dados e faz o *select*. Vou até selecionar aqui.

[06:11] Quando eu volto para a página e faço uma atualização da página *home*, várias vezes, era para termos várias consultas ao banco, mas não. Ele cacheou aquele conteúdo.

[06:22] Então agora, toda vez que eu tentar acessar e outros usuários tentarem acessar esse conteúdo dessa página, nossa aplicação não vai mais no banco de dados, ou seja, é um descanso para o MariaDB, esse cache.

[06:40] O que eu posso dizer a mais sobre isso é que cache é muito simples. Essa versão funciona bem, mas ela é muito simples. Deem uma olhada nessas outras anotações e o que elas significam, que é bem legal. Eu não vou me aprofundar mais sobre cache porque existem pelo menos dois cursos na Alura que falam bem sobre isso.

[07:00] Um específico de JPA - que é exatamente o nosso cenário aqui, estamos fazendo a consulta ao banco de dados usando cache. É muito bom esse treinamento e fala sobre cache, sobre `ehcache`. E tem outro sobre Spring Boot, que também fala sobre cache em outro cenário.

[07:17] E como nosso curso é sobre MVC, não vamos entrar em detalhes, mas é legal que você conheça e entenda como é fácil de plugar, encaixar na aplicação e ter uma maneira boa de você raciocinar ali: "Estou fazendo essa funcionalidade, eu preciso dar uma otimizada, eu preciso ter essa preocupação, preciso paginar, ordenar e cachear".

[07:40] Por isso que eu falei sobre isso nesse treinamento, porque é importante você ter essa cabeça de tentar entender como as nossas páginas vão ser usadas, como o nosso sistema vai ser usado e como prepará-lo para entrar em produção com saúde. Aguentando as consultas e tudo mais.

[08:02] Vamos continuar no próximo vídeo. Até lá!