05

Criando o pool de conexões

Transcrição

[00:00] Fala, aluno. Tudo bom? Agora já com o conhecimento adquirido de como funciona um Pool de conexões, nós vamos partir de fato para a implementação na nossa aplicação. Hoje, o que nós fazemos é retornar uma conexão seca, então quando é feita a requisição que precisa recuperar uma conexão, ela vai recuperar a conexão com o nosso banco de dados, vai fechar e quando vier outra requisição, ela vai abrir uma nova requisição.

[00:27] Nós vimos que isso, com o tempo, com aplicações de grande complexidade, com processamento de muitos dados, isso pode ficar complicado no futuro, porque pode onerar o nosso banco de dados. Então para conseguirmos implementar o nosso Pool de conexões, nós vamos ter que inserir dois JARs externos à nossa aplicação clicando em "loja-virtual-repository" com o botão direito e selecionando a opção "Properties" para abrir a janela "Properties for loja-virtual-repository".

[00:52] Selecionando "Classpath" e clicando no botão "Add External JARs...", esses arquivos serão o "mchange-commons-java-0.2.16" e o "c3p0-0.9.5.4". Esses dois JARs, eles estarão disponíveis no repositório da Alura para vocês baixarem. Eu peço encarecidamente que vocês usem esses dois JARs, que estão no repositório, porque são os que estamos usando aqui na aula.

[01:13] Podemos utilizar um mais novo? Pode, só que pode ter problema de compatibilidade, o nosso problema pode ser por conta de versão, e essa não é a intenção. A intenção aqui é praticarmos esses conhecimentos que estamos

adquirindo do JDBC. Então vamos evitar utilizar outra versão, vamos usar as que estão no repositório, que vai dar tudo certo.

[01:38] Vou aplicar aqui os arquivos. Então, como foi falado, na aula em que nós fizemos o desenho de como funciona um Pool de conexões, a ideia é que agora eu não fique abrindo e fechando conexões com o meu banco de dados. Mas também eu tenho que ter um meio termo, também não posso enfileirar as minhas conexões e esperar que eu só tenha uma conexão.

[02:01] Então vamos ter que criar esse Pool de conexões mesmo. E qual é a ideia? Toda vez agora, no meu `public ConnectionFactory`, toda vez que ele for instanciado, eu quero que o meu Pool de conexões, ele seja instanciado também, que ele me dê um Pool de conexões, na verdade.

[02:19] Então, com a C3P0, eu tenho um `comboPooledDataSource`. Esse `comboPooledDataSource`, nós vamos instanciar ele também para podermos fazer algumas configurações nele. Então deixa eu fazer: `ComboPooledDataSource comboPooledDataSource = new ComboPooledDataSource();`.

[02:48] Antes de continuarmos, para não ficar tão obscuro para nós o porquê utilizar esse JAR, essa biblioteca, nós vamos no site da biblioteca C3P0, que é esse "mchange.com/project/c3p0/". Ela vai explicar que o C3P0, ele é uma *library*, é um JAR que faz o que os drivers JDBC, hoje eles fazem.

[03:19] Então, por exemplo, eu falei para vocês que o SQL Server vai ter uma implementação para Datasource, o Postgre vai ter um driver que vai ter uma implementação para o Datasource. No caso do MySQL, precisamos utilizar esse C3P0 para implementar o nosso Pool de conexões para ser exposto pela nossa interface Datasource, então esse é o principal objetivo do C3P0, por isso estamos utilizando ele.

[03:52] Então vamos lá, continuando a configuração do nosso Pool, eu tenho que *setar* a `(jdbcUrl)`, que nada mais é do que a nossa string de conexão que já tínhamos usado anteriormente, quando estávamos retornando a conexão

com o Driver manager. Então vamos só copiar a URL e colar dentro dos parênteses de `comboPooledDataSource.setJdbcUrl();` .

[04:19] Nós temos também que *setar* o usuário com `comboPooledDataSource.setUser("");` , vai ser o `("root")` . E vamos *setar* o nosso password, com `comboPooledDataSource.setPassword("");` , que é `("root")` também. Quando nós estávamos na aula do desenho, nós falamos que nós configuramos o Pool de conexões e quem expõe ela, temos que implementar uma interface Datasource para poder expor essas informações para a nossa aplicação.

[05:00] Então o que nós vamos fazer aqui é criar uma variável que vai ser do tipo Datasource. Esse Datasource é como foi falado anteriormente, ele não é de um driver específico, igual é o `ComboPooledDataSource` , ela é do Java, então o Datasource é uma interface que será implementada por esses drivers, que têm por objetivo expor mesmo essas informações do Pool.

[05:32] Então, quando eu crio esse `public DataSource dataSource;` , eu posso fazer o seguinte, eu posso falar `this.dataSource =` recebe o `= ComboPooledDataSource;` , que eu estou falando o seguinte: eu configurei aqui o meu Pool de conexões e agora, Datasource, expõe isso para a minha aplicação. Então é a nossa Datasource que vai fazer com que o nosso Pool, ele funcione, ele seja criado de acordo com aquilo que vimos no nosso desenho.

[SCREENSHOTS]

[06:09] Uma vez que eu tenho esse `ComboPooledDataSource` configurado, eu já não vou mais querer prover uma conexão com o `return DriverManager` , eu vou então tirar essa parte do código e o que eu vou fazer é o seguinte: vou chamar o meu `return this.dataSource` . Dentro desse `DataSource` , se nós entrarmos nele, nós vamos ver que ele tem um `getConnection()` .

[06:32] Então eu estou falando o seguinte: pronto `DataSource` , pega para mim conexão que está disponível no Pool de conexões. Então com isso nós

começamos a ter mais ou menos aquela ideia de como funciona o desenho. Então vai vir uma requisição, ele vai pedir uma conexão, e essa conexão, ela vai estar disponível naquele Pool de conexões, eu não vou precisar abrir uma conexão direta com o nosso banco de dados.

[07:02] Então essa é a principal diferença para a nossa aplicação agora. Então em vez de eu ir no banco de dados eu ir direto depois que acabar o processamento da minha requisição, eu mato essa conexão? Não, eu vou reaproveitar essa conexão, uma que já esteja aberta, e quando acabar o processamento, a próxima requisição que chegar, a anterior estará aberta. Então essa é a ideia de utilizarmos o Pool de conexões e a interface Datasource.

[07:37] Vamos só tirar esse `import java.sql.DriverManager`, que não será utilizado. Agora vamos na classe `TestaConexao` e nós vemos que não mudou nada para nós, continuamos compilando do mesmo jeito. Só que agora com a diferença que quando eu instancio uma `ConnectionFactory`, eu estou instanciando, estou criando o meu Pool de conexões.

[08:02] Depois, quando eu uso `.recuperarConexao()`, eu estou só estou pegando a conexão que está no meu Pool, então é essa a diferença, e que faz total diferença em uma aplicação que processa milhares de registros, enfim, essa mudança faz toda a diferença. Se nós executarmos essa classe `TestaConexao`, nós vamos ver que vamos ter o mesmo cenário que nós tínhamos anteriormente.

[08:31] E o motivo de trazermos esse JAR, o "mchange-commons-java-0.2.16", é que ele é quem vai *logar*, referente ao Datasource, então ele vai trazer as configurações que estão atualmente no Datasource, porque temos outras configurações que podemos fazer. Nós não vamos nos aprofundar nessa aula, mas aqui estamos falando mais ou menos isso: inicializou o Pool, e foi botando algumas informações, que depois, com o tempo, vamos entendendo.

[09:07] Na documentação vai ter sobre todas essas configurações que são possíveis fazer com o C3P0, nós não podemos nos aprofundar aqui, porque sao

muitas configurações, mas enfim, é basicamente para isso que ele serve. Então agora, para termos um exemplo mais real, vamos recuperar a nossa listagem do banco de dados, que vamos ver também que ela continua do mesmo jeito.

[09:35] Então nós temos agora aqueles dois registros que nós sempre estávamos usando ao longo do curso. Então, aparentemente, o nosso Pool, ele está configurado corretamente, estamos recuperando a conexão do Pool de conexões, por isso estamos conseguindo visualizar as informações do banco.

[09:54] Só que agora temos algumas maneiras de trabalhar com esse Pool, que é *setar* o tamanho do Pool, porque eu não quero deixar livre. A nossa intenção ao usar o Pool é justamente para termos um controle melhor das nossas conexões, então nas próximas aulas, vamos conseguir melhorar essas nossas configurações, porque vamos evitar os problemas de termos várias conexões abertas, enfim.

[10:23] Então hoje, nesta aula, estamos tratando sobre a implementação do Pool, porque foi algo que vimos apenas no desenho, então agora a ideia é que vamos de fato vendo os benefícios desse Pool de conexões. Então ficamos por aqui, espero que vocês tenham gostado e até a próxima aula.