05. Escalabilidade com pool de conexões

Date	@13/12/2022
Categoria	Java e persistência
Curso	Java e JDBC: trabalhando com um banco de dados

Tópicos

- Projeto da aula anterior
- Download dos JARs
- O que é pool e datasource?
- Um único cliente
- Criando o pool de conexões
- Diversos clientes
- Testando o pool de conexões
- Pool com conexões ocupadas
- Faça que eu fiz
- O que aprendemos?

Criando o pool de conexões

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.

```
import java.sql.Connection;
import java.sql.SQLException;


public class ConnectionFactory{

    public DataSource dataSource;

    public ConnectionFactory(){
        ComboPooledDataSource comboPooledDataSource = new ComboPooledDataSource();
        comboPooledDataSource.setJdbcUrl("jdbc:mysql://localhost/loja_virtual?useTimezone=true&serverTimezone=UTC","root","root");
        comboPooledDataSource.setUser("root");
        comboPooledDataSource.setPassoword("root");

        this.dataSource = comboPooledDataSource;
    }

    public Connection recuperarConexao() throws SQLException{
        return this.dataSource.getConnection();
    }
}
```

Diversos clientes

Em um cenário onde diversos clientes podem acessar uma mesma aplicação simultaneamente, qual a abordagem mais apropriada?

- Reciclar um conjunto de conexões de tamanho fixo ou dinâmico
  - Essa é a abordagem do *pool de conexão.* Vamos abrir uma quantidade de conexões e reaproveitá-las.

Testando o pool de conexões

Nessa parte da aula o professor mostra como definimos um tamanho do nosso pool de conexões, e para fazer isso foi utilizado o comando:

```
comboPooledDataSource.setMaxPoolSize(tamanhoMaximoDoPoolDeConexoes);
```

Pool com conexões ocupadas

Em um *pool* simples, com 9 conexões e todas elas ocupadas, o que acontece quando o 10º usuário se conecta?

- O 10º usuário esperará alguma das 9 conexões ficar disponível
  - Assim que for disponibilizada, o 10º cliente terá sua requisição processada.

O que aprendemos?

Nesta aula, aprendemos que:

- É boa prática usar um ***pool de conexões***
- Um *pool* de conexões administra/controla a quantidade de conexões abertas
  - Normalmente tem um mínimo e máximo de conexões
- Como existe uma interface que representa a conexão(`java.sql.Connection`), também existe uma interface que representa o *pool* de conexões (`javax.sql.DataSource`)
- **C3P0** é uma implementação Java de um *pool* de conexão
