**T05 – Estabelecendo conexões**

Igor Augusto Reis Gomes – 12011BSI290 – [igor.augusto@ufu.br](mailto:igor.augusto@ufu.br)

Heitor Guimarães Da Fonseca Filho – 12011BSI203 – [heitor.filho@ufu.br](mailto:heitor.filho@ufu.br)

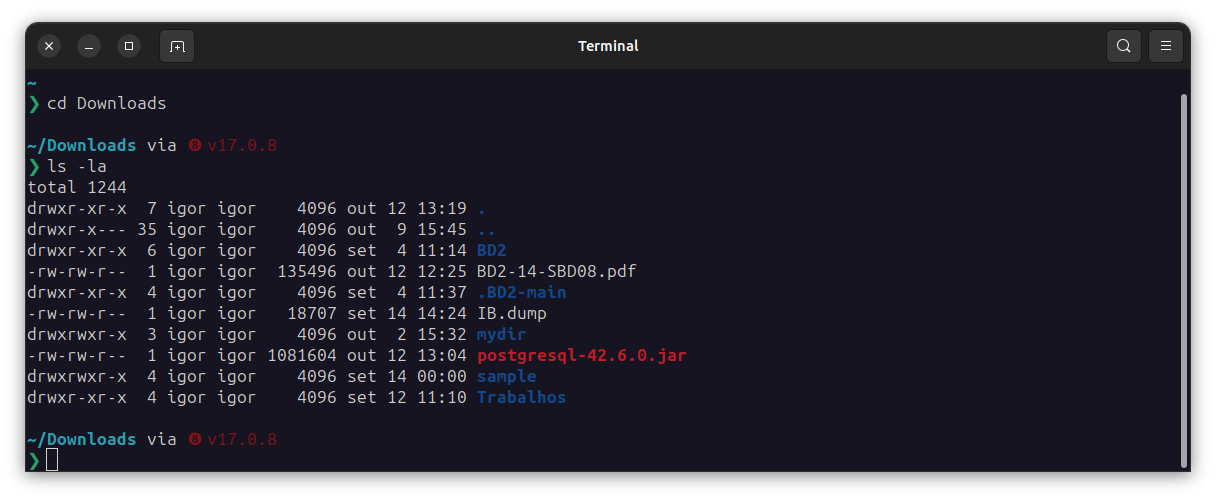
1. **Na aba Serviços do NetBeans, opção Drivers, você tanto pode utilizar os drivers mapeados por padrão quanto se conectar a um novo banco de dados. No exemplo abaixo foram feitas duas conexões ao banco de testes do tutorial e uma conexão a um banco do SGBD PostgreSQL**

Utilizei o Datagrip:

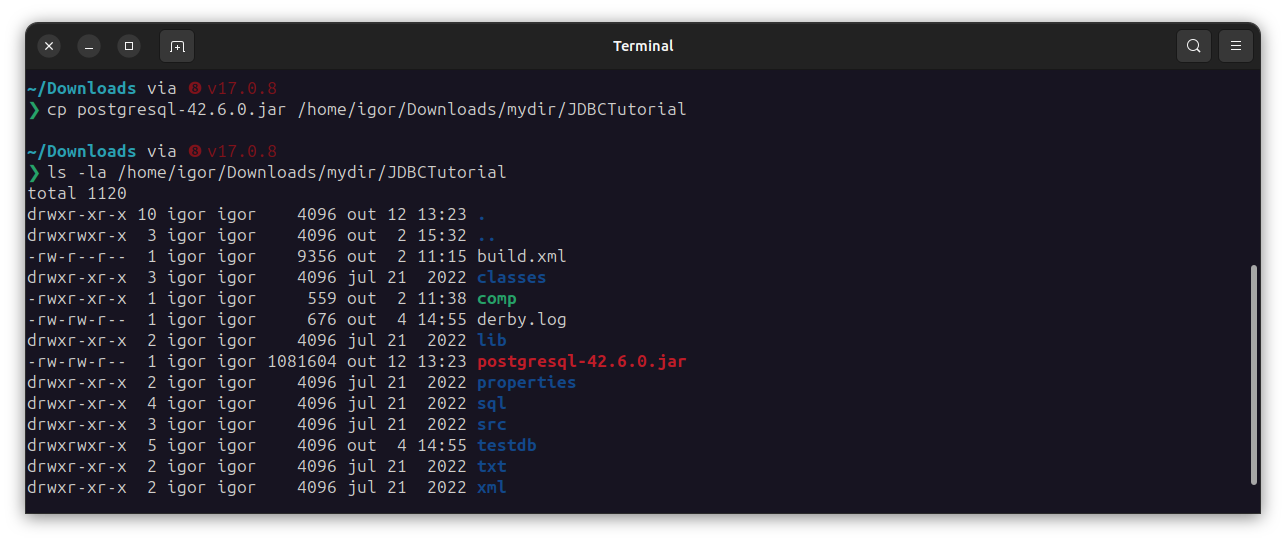
**Tela de computador

Descrição gerada automaticamente**

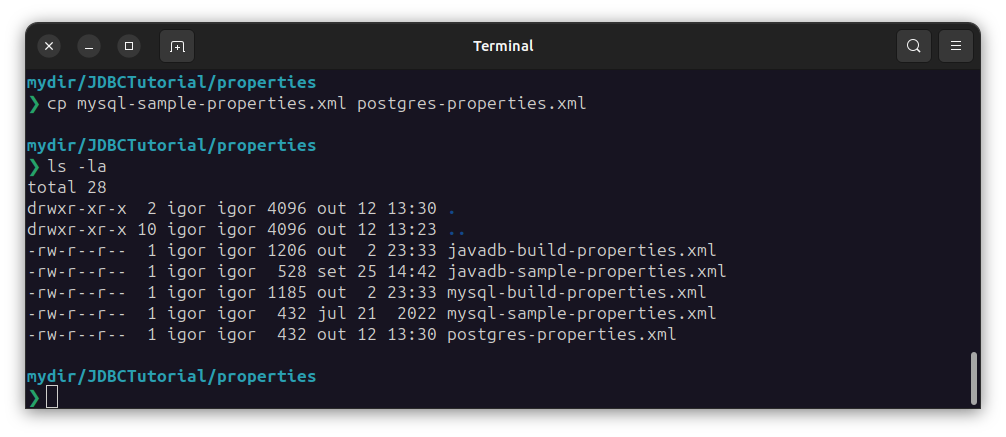
1. **Baixe o arquivo postgresql-42.2.4.jar do moodle. Depois abra o terminal de comandos do linux e posicione-se no diretório em que baixou o arquivo postgresql-42.2.4.jar para esta aula. Para que não se recorda: use para isso o comando cd;**



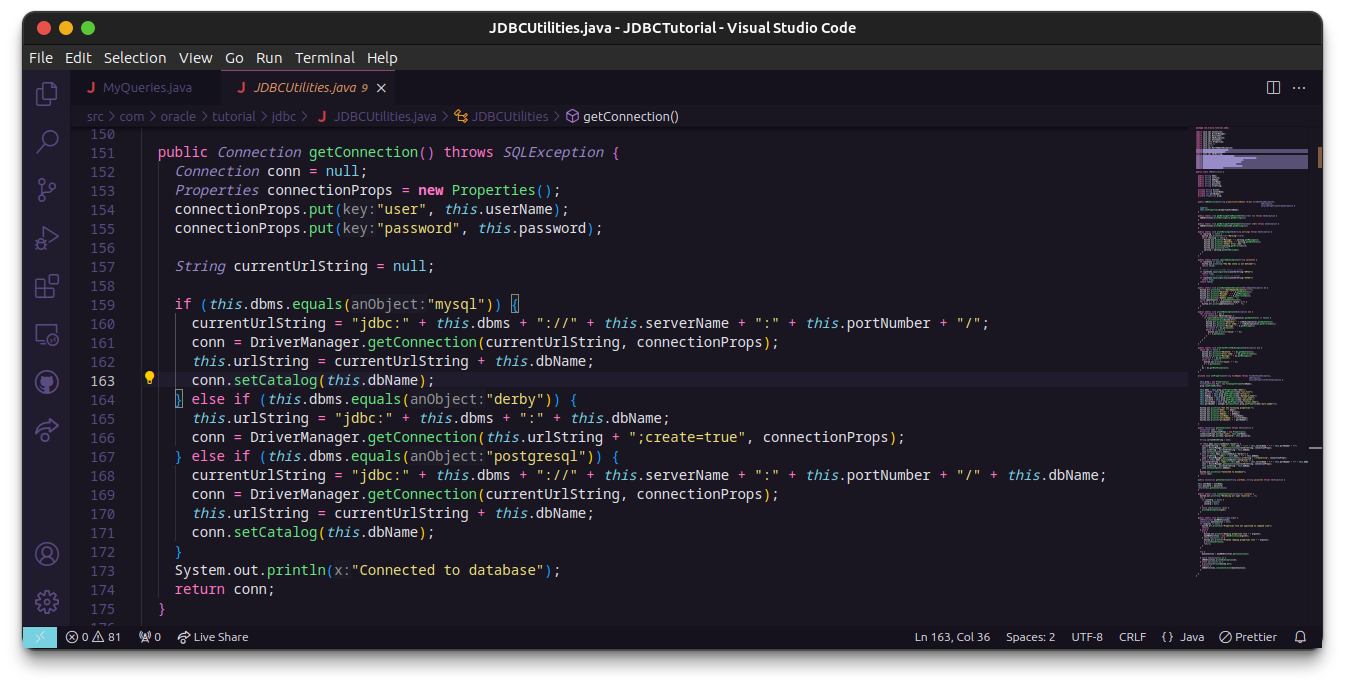
1. **Copie o driver do postgres de nome postgresql-42.2.4.jar para a pasta raiz do tutorial do java digitando o comando cp: cp postgresql-42.2.4.jar /home/myuser/mydir/JDBCTutorial/**



1. **Faça uma cópia do arquivo mysql-sample-properties.xml, que está na pasta properties, com o nome de postgres-properties.xml. Esse novo arquivo será utilizado para se conectar ao banco de dados do postgres. Terminada a cópia: edite o novo arquivo com os parâmetros corretos, lembre-se do usuário e senha do usuário postgres e da porta de conexão padrão. Se você não estiver conseguindo se lembrar, tente criar uma conexão usando, por exemplo, o NetBeans. Parâmetros iguais devem ser configurados no arquivo postgres-properties.xml.**

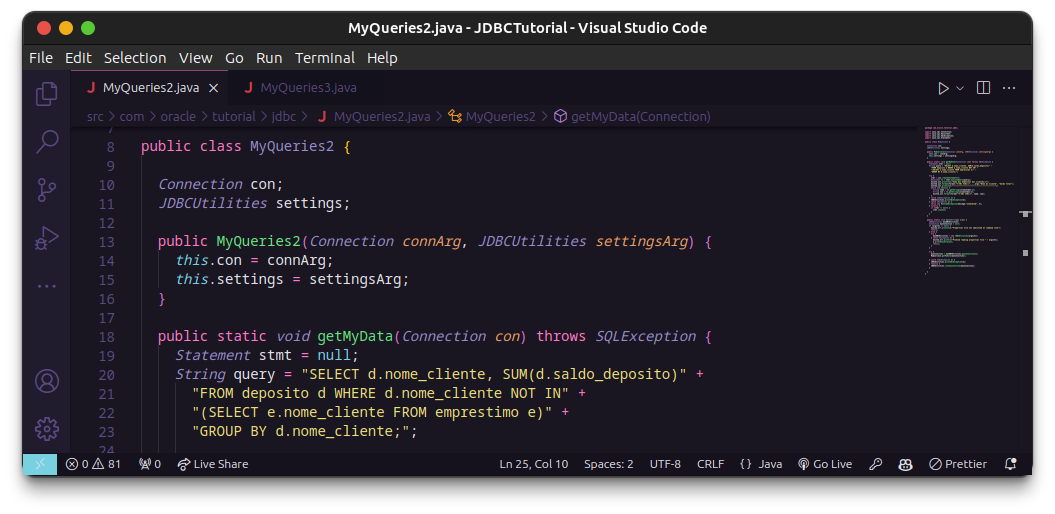


1. **Modifique o programa JDBCUtilities.java para incorporar estas linhas que o habilitam a se conectar a um banco de dados postgres. Veja que no código existem duas funções com uma estrutura if para se conectar ao derby e ao mysql. Altere ambas as funções acrescentando este código:**



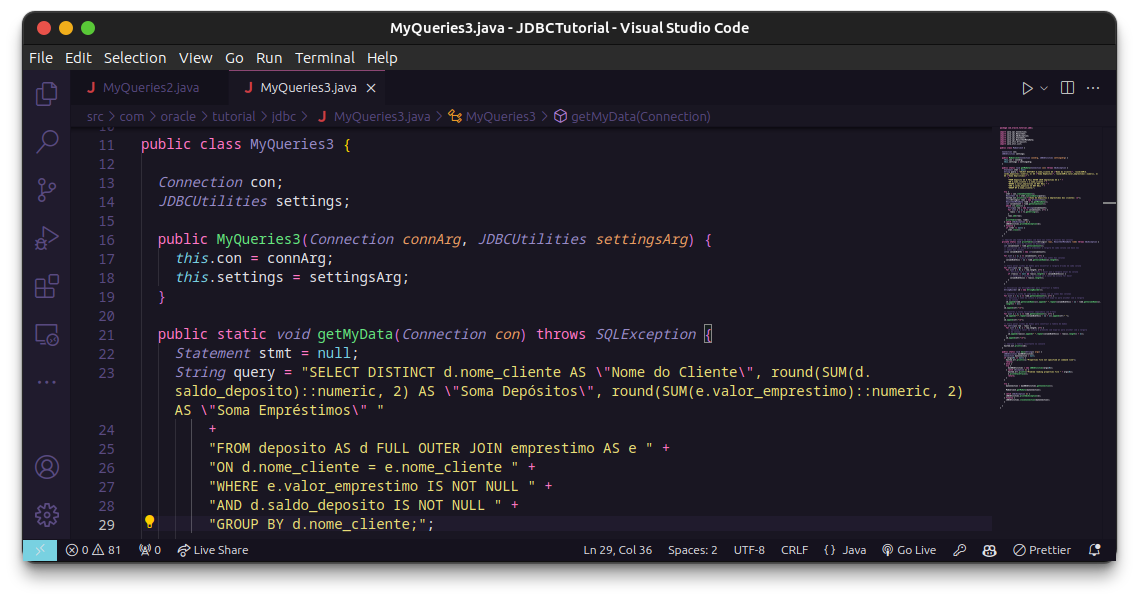
1. **Crie duas novas classes a partir da classe MyQueries.java (MyQueries2.java e MyQueries3.java) para realizar as seguintes consultas e exibir os resultados na tela da linha de comando. Para esse fim é necessário tanto mudar o nome do arquivo quanto das referências à classe que está definida dentro do arquivo ou a Java vai acusar erro de compilação. Estas são as consultas que os novos programas devem fazer:**
   1. Retorne os nomes de todos os clientes que possuem apenas depósitos, bem como a soma de depósitos de cada cliente (Implementar no MyQueries2.java);

Nessa consulta, o subselect retorna os nomes dos clientes que possuem empréstimos e o select principal retorna os nomes dos clientes que não estão na lista retornada pelo subselect.

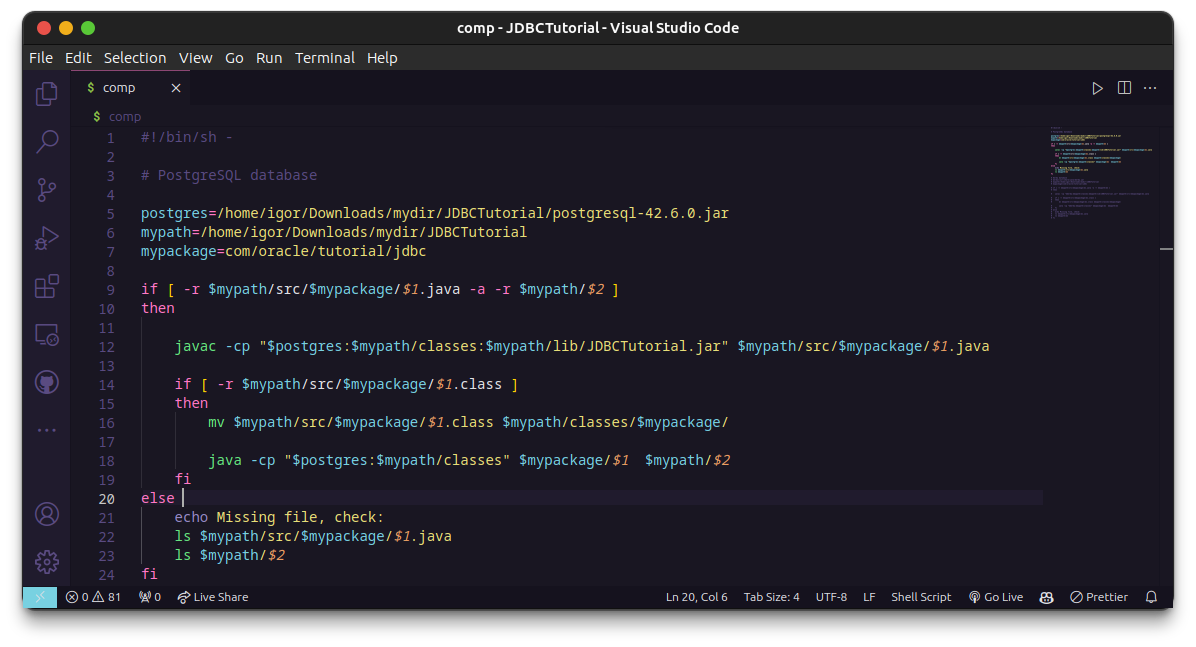


* 1. Retorne os nomes dos clientes que possuem depósitos e empréstimos (ambos) com as respectivas somas (Implementar no MyQueries3.java);

Já nessa consulta, utilizei “Distinct” para não repetir os nomes dos clientes e o “full outer join” para que apareçam os clientes que possuem apenas depósitos ou apenas empréstimos. Adicionalmente, também utilizei alias para os nomes das colunas, e fiz uso do “round” para arredondar os valores (para duas casas) em conjunto com o “::numeric” para converter os valores para o tipo “numeric” (para que o round funcione).

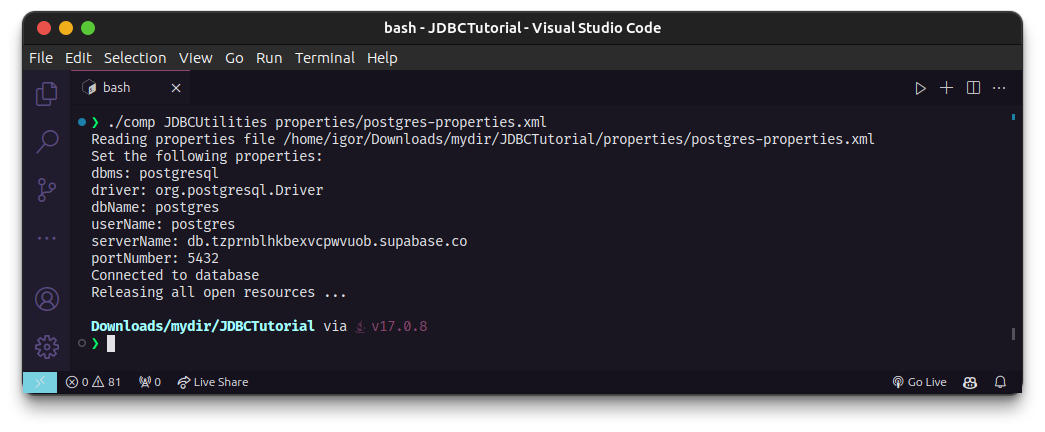


1. **Modifique o arquivo 'comp' da aula anterior para criar uma variável postgres com a configuração adequada à estrutura de diretórios de sua máquina. Use a variável derby como exemplo de como deve ficar a variável postgres dentro do arquivo comp.**
   1. postgres=/home/myuser/mydir/JDBCTutorial/postgresql-42.2.4.jar
   2. Acrescente a variável $postgres no parâmetro -classpath (-cp) ao acionar o compilador java (denominado javac) e também ao acionar a máquina virtual java (denominado java). Modifique o script comp para este fim.
   3. Lembre-se que se o arquivo comp foi copiado agora e ainda não utilizado então ele ainda não tem permissão de execução. Neste caso o comando chmod 755 comp vai habilitá-lo para este fim;



1. **Compile primeiro a classe JDBCUtilities.java que foi modificada nas etapas anteriores: ./comp JDBCUtilities properties/javadb-sample-properties.xml**

Utilizei como banco o serviço [Supabase](https://supabase.com/), uma alternativa que achei mais adequada ao banco local:



1. **Compile e execute em seguida as duas classes MyQueries2 e MyQueries3.**

**Lembre-se que o novo arquivo postgres-properties.xml (passo 4) deve ser passado para todos os programas compilados e executados com o programa comp.**

Fiz uma tabela básica na primeira consulta, e uma mais elaborada (com uma função) na segunda:

