

AT05 – Estabelecendo Conexões

Layza Nauane – BD2 – 12211BSI251

1. Introdução

O objetivo desse trabalho é estabelecer uma conexão JDBC ao banco de dados postgres.

2. Consultas SQL e Execuções

Tenho duas classes java e preciso implementar duas consultas, uma em cada classe, sendo elas:

MyQueries2.java:

- *Retorne os nomes de todos os clientes que possuem apenas depósitos, bem como a soma depósitos de cada cliente.*

Função com a query:

```
public static void getMyData(Connection con) throws SQLException {
    Statement stmt = null;
    String query =
        "SELECT A.NOME_CLIENTE AS NOME, SUM(A.SALDO_DEPOSITO) AS TOTAL_DEPOSITO " +
        "FROM DEPOSITO A LEFT JOIN CLIENTE B ON A.NOME_CLIENTE = B.NOME_CLIENTE " +
        "WHERE B.NOME_CLIENTE NOT IN (SELECT DISTINCT NOME_CLIENTE FROM EMPRESTIMO) " +
        "GROUP BY A.NOME_CLIENTE";

    try {
        stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery(query);
        System.out.println("\n--- Clientes que possuem depositos e seus respectivos saldos: ---\n");
        while (rs.next()) {
            String clientName = rs.getString("NOME");
            float sumDeposits = rs.getFloat("TOTAL_DEPOSITO");
            System.out.println("Nome: " + clientName + ", Saldo total depositos: " + sumDeposits + "\n");
        }
    } catch (SQLException e) {
        JDBCUtilities.printSQLException(e);
    } finally {
        if (stmt != null) { stmt.close(); }
    }
}
```

Execução:

```
user@user-Latitude-E7450: ~/Documentos/JDBCTutorial2
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$ chmod 755 comp
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$ ./comp JDBCUtilities properties/javadb-sample-properties.xml
Reading properties file /home/user/Documentos/JDBCTutorial2/properties/javadb-sample-properties.xml
Set the following properties:
dbms: derby
driver: org.apache.derby.jdbc.EmbeddedDriver
dbName: testdb
userName:
serverName: localhost
portNumber: 3306
Connected to database
Releasing all open resources ...
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$ ./comp MyQueries2 properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB
userName: postgres
serverName: localhost
portNumber: 5432
Connected to database

--- Clientes que possuem depositos e seus respectivos saldos: ---
Nome: Cláudia Santos Mota, Saldo total depositos: 3599.29
Nome: Thiago Andrade Fluzza, Saldo total depositos: 479.66
Nome: Wantuil Diniz e Souza, Saldo total depositos: 84.17
Nome: Alexandre Marcio de Souza, Saldo total depositos: 1201.37
Nome: Felipe Peres Bichara Junior, Saldo total depositos: 7417.87
Releasing all open resources ...
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$
```

MyQueries3.java:

- Retorne os nomes dos clientes que possuem depósitos e empréstimos (ambos) com as respectivas somas.

Função com a query:

```
public MyQueries3(Connection connArg, JDBCUtilities settingsArg) {
    this.con = connArg;
    this.settings = settingsArg;
}

public static void getMyData(Connection con) throws SQLException {
    Statement stmt = null;
    String query =
        "SELECT B.NOME_CLIENTE AS NOME, SUM(B.SALDO_DEPOSITO) AS TOTAL_DEPOSITO, SUM(A.VALOR_EMPRESTIMO) AS TOTAL_EMPRESTIMO "
        + "FROM EMPRESTIMO A "
        + "JOIN DEPOSITO B ON A.NOME_CLIENTE = B.NOME_CLIENTE "
        + "GROUP BY B.NOME_CLIENTE";

    try {
        stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery(query);
        System.out.println("\n--- Clientes que possuem depositos e emprestimos com seus respectivos saldos: ---\n");
        while (rs.next()) {
            String clientName = rs.getString("NOME");
            float sumDeposits = rs.getFloat("TOTAL_DEPOSITO");
            float sumLoans = rs.getFloat("TOTAL_EMPRESTIMO");
            System.out.println("Nome: " + clientName + ", Saldo total depositos: " + sumDeposits + ", Saldo total emprestimos: "
            );
        }
    } catch (SQLException e) {
        JDBCUtilities.printSQLException(e);
    } finally {
        if (stmt != null) { stmt.close(); }
    }
}
```

Execução:

```
user@user-Latitude-E7450: ~/Documentos/JDBCTutorial2
Releasing all open resources ...
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$ ./comp MyQueries3 properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB
userName: postgres
serverName: localhost
portNumber: 5432
Connected to database

--- Clientes que possuem depositos e emprestimos com seus respectivos saldos: ---
Nome: Germano Luiz de Paula, Saldo total depositos: 209635.44, Saldo total emprestimos: 1.6757003E8
Nome: Bruno Miranda Pacheco de Castro, Saldo total depositos: 2512.88, Saldo total emprestimos: 1139.08
Nome: Joaquim Carlos Reis, Saldo total depositos: 997.12, Saldo total emprestimos: 805.88
Nome: Thiago Leôncio Guinãres, Saldo total depositos: 3506.84, Saldo total emprestimos: 1411.99
Nome: Clayton Pereira Bonfim, Saldo total depositos: 3231.84, Saldo total emprestimos: 1792.56
Nome: Everardo Monfort Leitão, Saldo total depositos: 3562.54, Saldo total emprestimos: 3773.96
Nome: Andre Cabral da Silva, Saldo total depositos: 8980.94, Saldo total emprestimos: 5225.96
Nome: Carolina Soares, Saldo total depositos: 3516.28, Saldo total emprestimos: 1324.74
Nome: Marco Aurélio Santos, Saldo total depositos: 9493.32, Saldo total emprestimos: 3775.64
Nome: Marcos Andrade, Saldo total depositos: 3493.63, Saldo total emprestimos: 886.85
Nome: Gilmar Negreiros Carvalho, Saldo total depositos: 10230.12, Saldo total emprestimos: 4234.1
Releasing all open resources ...
user@user-Latitude-E7450:~/Documentos/JDBCTutorial2$
```

3. Configurações e Modificações

Para o trabalho de hoje foi necessário baixar um novo arquivo .jar e realizar algumas modificações em alguns arquivos da pasta raiz JDBCTutorial.

Passo 1:

- Baixar o arquivo *postgresql-42.2.4.jar*

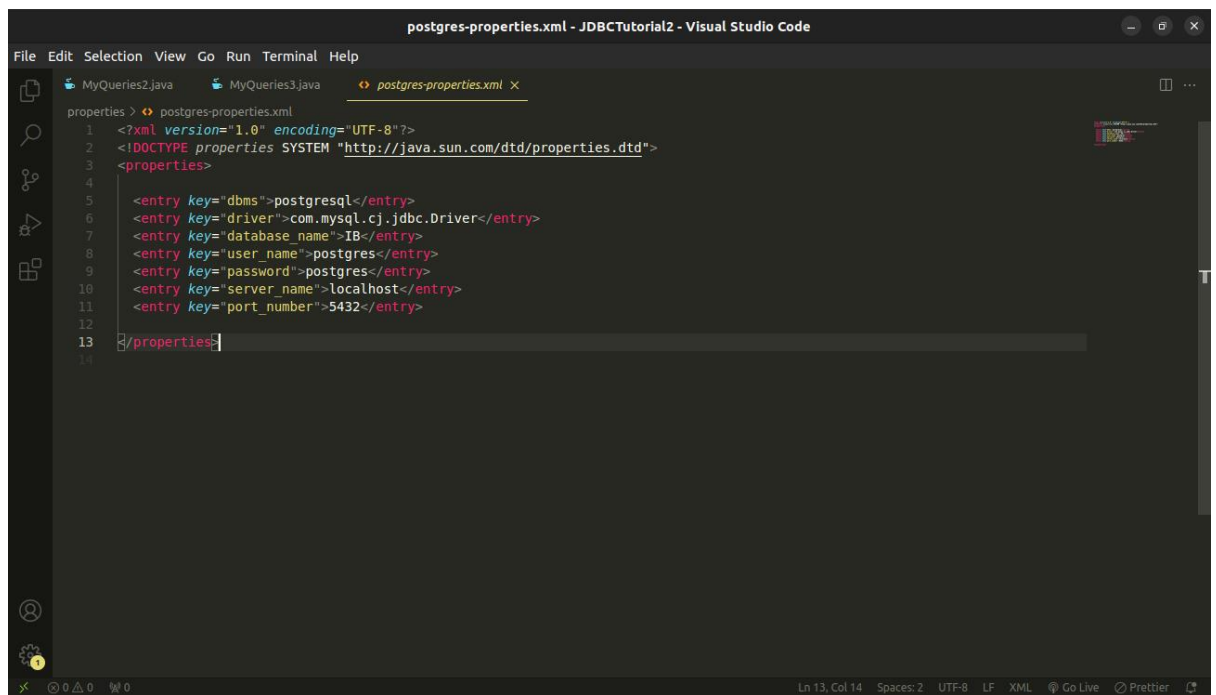
Passo 2:

- Copiar o drive do postgres (*postgresql-42.2.4.jar*) para a pasta raiz JDBCTutorial com o comando cp:

```
cp postgresql-42.2.4.jar /home/user/Documentos/JDBCTutorial/
```

Passo 3:

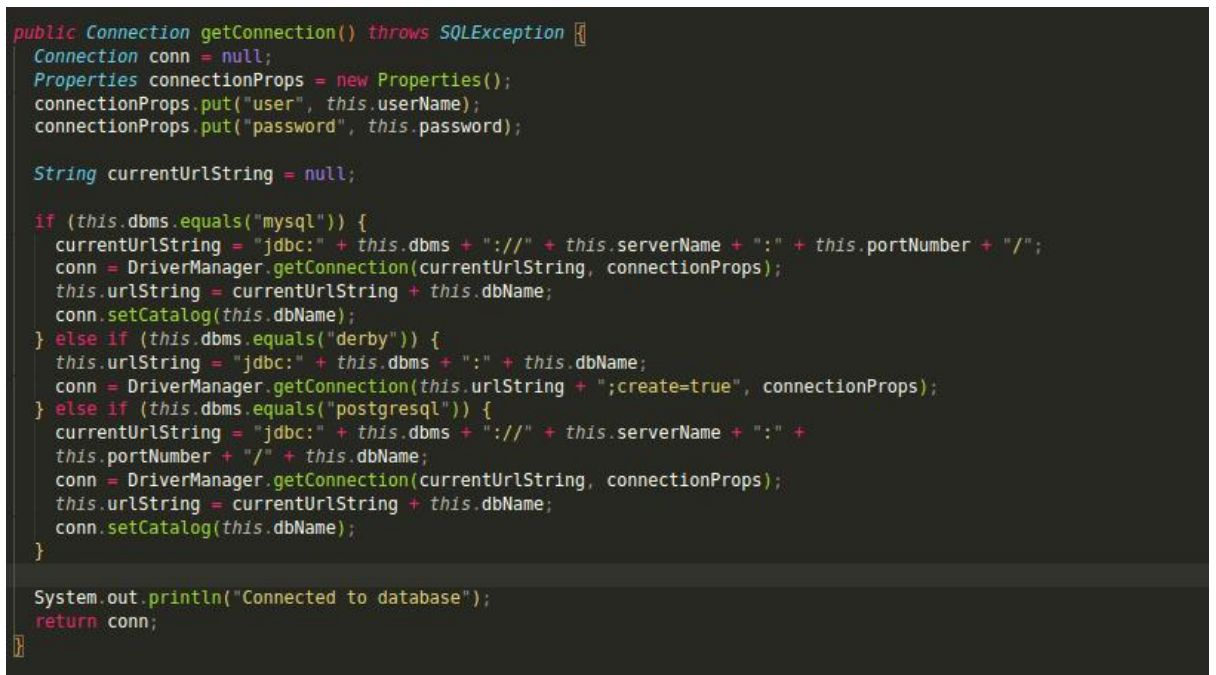
- Copiar o arquivo *mysql-sample-properties.xml*, da pasta *properties* e renomear a cópia para *postgres-properties.xml*. Esse novo arquivo irá nos conectar ao banco de dados do postgres, então é preciso modificar alguns parâmetros.



```
postgres-properties.xml - JDBCtutorial2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
MyQueries2.java MyQueries3.java postgres-properties.xml
properties > postgres-properties.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
3 <properties>
4
5   <entry key="dbms">postgresql</entry>
6   <entry key="driver">com.mysql.cj.jdbc.Driver</entry>
7   <entry key="database_name">IB</entry>
8   <entry key="user_name">postgres</entry>
9   <entry key="password">postgres</entry>
10  <entry key="server_name">localhost</entry>
11  <entry key="port_number">5432</entry>
12
13 </properties>
14
```

Passo 4:

- Modificar o programa *JDBCUtilities.java*, adicionando algumas linhas que o habilitaram a se conectar com o banco de dados. No código há duas funções, uma para se conectar ao derby e outra ao mysql, vamos atualizar as duas com o trecho de código abaixo:



```
public Connection getConnection() throws SQLException {
    Connection conn = null;
    Properties connectionProps = new Properties();
    connectionProps.put("user", this.userName);
    connectionProps.put("password", this.password);

    String currentUrlString = null;

    if (this.dbms.equals("mysql")) {
        currentUrlString = "jdbc:" + this.dbms + "://" + this.serverName + ":" + this.portNumber + "/" + this.dbName;
        conn = DriverManager.getConnection(currentUrlString, connectionProps);
        this.urlString = currentUrlString + this.dbName;
        conn.setCatalog(this.dbName);
    } else if (this.dbms.equals("derby")) {
        this.urlString = "jdbc:" + this.dbms + ":" + this.dbName;
        conn = DriverManager.getConnection(this.urlString + ";create=true", connectionProps);
    } else if (this.dbms.equals("postgresql")) {
        currentUrlString = "jdbc:" + this.dbms + "://" + this.serverName + ":" + this.portNumber + "/" + this.dbName;
        conn = DriverManager.getConnection(currentUrlString, connectionProps);
        this.urlString = currentUrlString + this.dbName;
        conn.setCatalog(this.dbName);
    }

    System.out.println("Connected to database");
    return conn;
}
```



```

public Connection getConnection(String userName, String password) throws SQLException {

    this.userName = userName;
    this.password = password;

    Connection conn = null;
    Properties connectionProps = new Properties();
    connectionProps.put("user", this.userName);
    connectionProps.put("password", this.password);

    String currentUrlString = null;

    if (this.dbms.equals("mysql")) {
        currentUrlString = "jdbc:" + this.dbms + "://" + this.serverName + ":" + this.portNumber + "/";
        conn = DriverManager.getConnection(currentUrlString, connectionProps);
        this.urlString = currentUrlString + this.dbName;
        conn.setCatalog(this.dbName);
    } else if (this.dbms.equals("derby")) {
        this.urlString = "jdbc:" + this.dbms + ":" + this.dbName;
        conn = DriverManager.getConnection(this.urlString + ";create=true", connectionProps);
    } else if (this.dbms.equals("postgresql")) {
        currentUrlString = "jdbc:" + this.dbms + "://" + this.serverName + ":" +
            this.portNumber + "/" + this.dbName;
        conn = DriverManager.getConnection(currentUrlString, connectionProps);
        this.urlString = currentUrlString + this.dbName;
        conn.setCatalog(this.dbName);
    }

    System.out.println("Connected to database");
    return conn;
}

```

Passo 5:

- Modificar o arquivo *comp* para criar uma variável *\$postgres*.

```

comp
1  #!/bin/sh -
2
3  derby=/usr/share/java/derby.jar
4  mypath=/home/user/Documentos/JDBCTutorial2
5  mypackage=/com/oracle/tutorial/jdbc
6  postgres=/home/user/Documentos/JDBCTutorial2/postgresql-42.2.4.jar
7
8  if [ -r $mypath/src/$mypackage/$1.java -a -r $mypath/$2 ]
9  then
10
11      javac -cp "$derby:$postgres:$mypath/classes:$mypath/lib/JDBCTutorial.jar" $mypath/src/$mypackage/$1.java
12
13      if [ -r $mypath/src/$mypackage/$1.class ]
14      then
15          mv $mypath/src/$mypackage/$1.class $mypath/classes/$mypackage/
16
17          java -cp "$derby:$postgres:$mypath/classes" $mypackage/$1 $mypath/$2
18      fi
19  else
20      echo Missing file, check:
21      ls $mypath/src/$mypackage/$1.java
22      ls $mypath/$2
23  fi
24
25

```