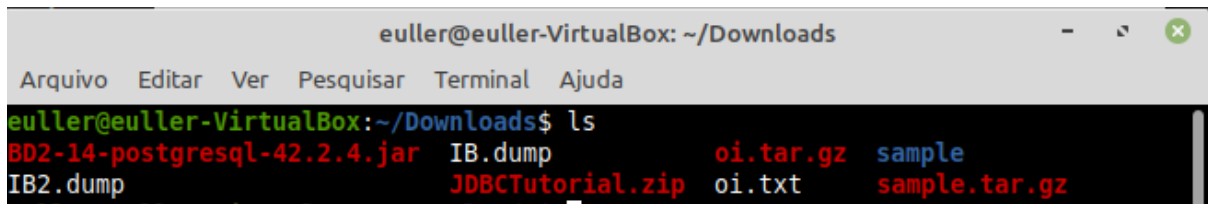


Universidade Federal De Uberlândia
Faculdade de Computação
Sistemas De Informação
Banco de dados 2
GSI021

EULLER HENRIQUE BANDEIRA OLIVEIRA
11821BSI210

T07-Criando e populando tabelas via JDBC

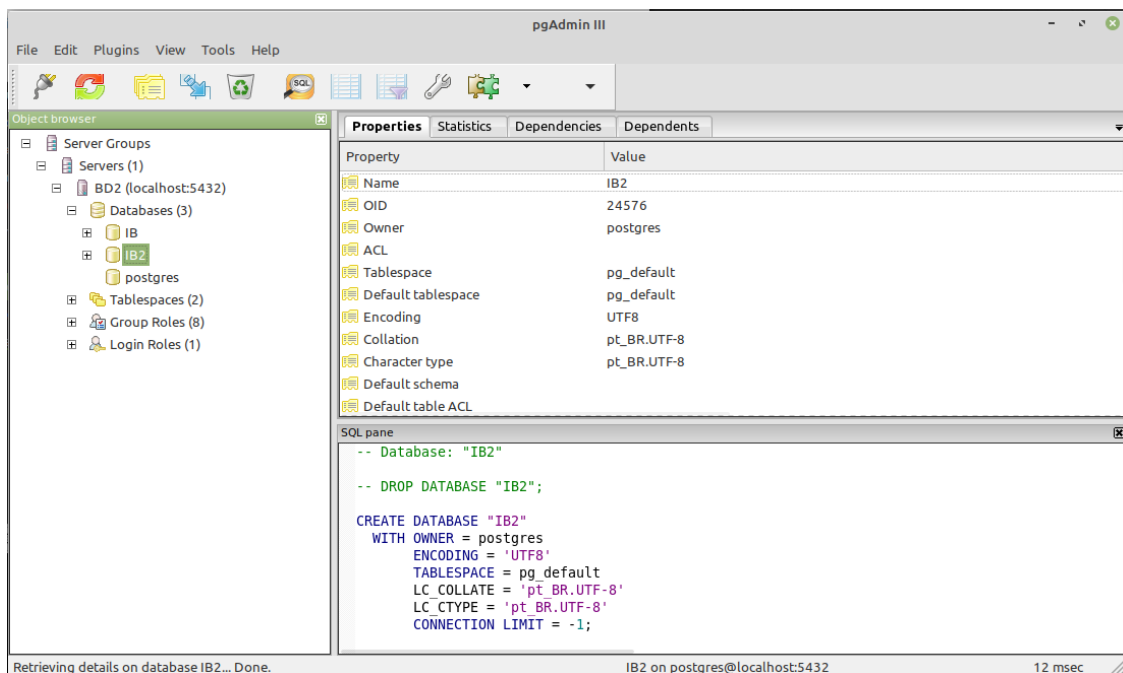
1) Nesta atividade vamos utilizar uma nova versão do banco da Instituição Bancária (IB). Não adianta tentar inserir estes dados no banco das aulas anteriores, visto que acontecerão erros de integridade referencial para todas as linhas da nova tabela que vamos criar e popular. Portanto, baixe o arquivo IB2.dump:
<https://www.dropbox.com/s/6zuffauqzvd4fde/IB2.dump>



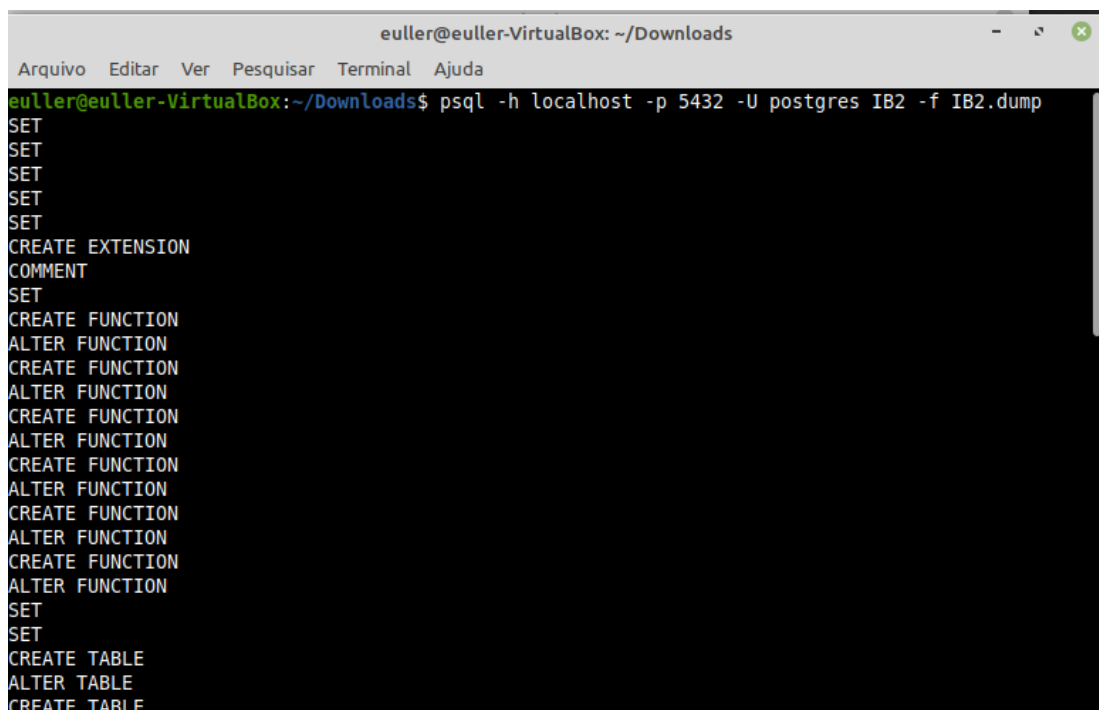
```
euller@euller-VirtualBox: ~/Downloads
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda

euller@euller-VirtualBox:~/Downloads$ ls
BD2-14-postgresql-42.2.4.jar  IB.dump          oi.tar.gz  sample
IB2.dump                     JDBCtutorial.zip oi.txt      sample.tar.gz
```

2) Crie um banco de dados vazio na interface do pgadmin3 de nome IB2;



3) Executar a carga do arquivo IB2.dump no banco IB2; 1. psql -h localhost -p 5432 -U postgres IB2 -f IB2.dump



```
euller@euller-VirtualBox: ~/Downloads
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda

euller@euller-VirtualBox:~/Downloads$ psql -h localhost -p 5432 -U postgres IB2 -f IB2.dump
SET
SET
SET
SET
SET
CREATE EXTENSION
COMMENT
SET
CREATE FUNCTION
ALTER FUNCTION
CREATE FUNCTION
ALTER FUNCTION
CREATE FUNCTION
ALTER FUNCTION
CREATE FUNCTION
ALTER FUNCTION
CREATE FUNCTION
ALTER FUNCTION
SET
SET
CREATE TABLE
ALTER TABLE
CREATE TABLE
```

4) Baixar o arquivo debito-populate-table.txt do link abaixo para a pasta raiz do tutorial do JDBC:

- 1. <https://www.dropbox.com/s/3nommkws4zqgvqgq/debito-populate-table.txt>

```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda

euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ ls
BD2-14-postgresql-42.2.4.jar  debito-populate-table.txt  src
build.xml                     derby.log                  testdb
classes                       JDBCUtilities.java        txt
comp                           lib                        xml
comp.save                     properties
comp.save.1                   sql
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$
```

- 2. Listar o conteúdo do arquivo baixado com cat ou editando-o com gedit;

debito-populate-table.txt							
~/Documentos/mydir/JDBCTutorial							
1	1001	900.75	4	2011-12-24	759	Pampulha	Pedro Alvares Sousa
2	1002	858.81	5	2012-11-18	34526	Central Marcos	Andrade
3	1003	353.28	5	2011-9-12	62195	Pampulha	Reinaldo Pereira da Silva
4	1004	1422.57	4	2012-7-15	45543	Gameleira	Elvis Fernando Da Silva
5	1005	1302.97	5	2012-3-2	22201	Central Joaquim	Carlos Reis
6	1006	268.32	3	2013-11-21	2370	Central Geraldo	Oliveira
7	1007	1199.2	4	2010-10-8	42436	Central Andrade	de Freitas
8	1008	963.43	1	2013-9-29	21707	Gameleira	Bruno Tadeu Pita
9	1009	1114.45	5	2011-11-23	46539	Central Alexandre	Márcio de Souza
10	1010	647.93	4	2011-7-18	68348	Cidade Jardim	Rafael Mesquita Gondim
11	1011	322.39	2	2011-4-13	57745	Cidade Jardim	Maria Raquel Teixeira de Castro
12	1012	354.2	3	2012-8-7	55878	Pampulha	Thiago Leôncio Guimarães
13	1013	219.93	2	2011-9-27	70612	Pampulha	Rafael Mesquita Gondim
14	1014	1081.85	5	2012-11-7	1767	Gameleira	Manuel Oliveira
15	1015	571.7	5	2013-4-5	76313	UFU	Bruno Miranda Pacheco de Castro
16	1016	572.33	2	2011-1-25	55207	Cidade Jardim	Marta Avelar Santos
17	1017	1462.73	3	2012-8-19	69118	Central João Boiadeiro	
18	1018	201.53	2	2011-8-21	72069	Central Andrade	de Freitas
19	1019	376.18	3	2011-6-14	20195	Cidade Jardim	Norton Saint Clair Silva
20	1020	1238.17	5	2010-7-4	74545	Pampulha	Maria Raquel Teixeira de Castro
21	1021	1320.77	5	2011-8-15	46248	UFU	Carla Soares Sousa
22	1022	804.21	1	2010-11-2	99737	Cidade Jardim	Pedro Alvares Sousa
23	1023	677.68	4	2011-4-6	51643	Glória	Felippe Peres Bichara Júnior
24	1024	145.83	2	2011-5-21	8072	Gameleira	Germano Luiz de Paula
25	1025	1251.78	2	2011-6-21	8275	Glória	Marco Aurélio Santos
26	1026	684.66	4	2013-9-19	17477	Glória	Gustavo Baer Albuquerque
27	1027	1227.99	5	2012-8-13	93125	Gameleira	André Cabral da Silva
28	1028	846.33	4	2011-2-25	8275	Glória	Marco Aurélio Santos
29	1029	448.13	5	2011-7-24	74545	Pampulha	Maria Raquel Teixeira de Castro
30	1030	1401.8	2	2013-5-20	50073	Cidade Jardim	André Cabral da Silva
31	1031	1454.12	6	2012-1-8	30273	Glória	Jefferson Oliveira
32	1032	584.67	5	2011-5-1	3682	UFU	Maria Lúcia Alves
33	1033	1298.39	4	2012-9-21	30273	Glória	Jefferson Oliveira
34	1034	553.1	4	2010-11-14	34787	UFU	Joaquim Carlos Reis
35	1035	263	3	2012-2-19	759	Pampulha	Pedro Alvares Sousa
36	1036	993.79	5	2011-8-8	759	Pampulha	Pedro Alvares Sousa
37	1037	52.65	4	2010-11-21	93125	Gameleira	André Cabral da Silva
38	1038	691.37	2	2012-11-3	70612	Pampulha	Rafael Mesquita Gondim
39	1039	1051.78	3	2010-11-22	3682	UFU	Maria Lúcia Alves
40	1040	1016.92	3	2013-1-4	26538	Cidade Jardim	Cláudia Santos Mota

Perceba que este arquivo possui centenas de linhas separadas por tabulação. A próxima tarefa compreende a leitura desses dados em um loop e execução de um comando insert into para cada linha e assim popular a tabela com estes dados. Para tanto disponibilizei um código de exemplo para ler dados tabulados de um arquivo e imprimir o resultado na tela. Façamos uma execução deste código para vê-lo em ação

5) Crie um arquivo vazio com o gedit, insira o seguinte código no arquivo vazio e salve-o com o nome de ReadFile.java, na mesma pasta na qual o arquivo debito-populate-table.txt foi salvo:

```
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;
import java.util.Scanner;
public class ReadFile {
    public static void main(String[] args) throws IOException {
        BufferedReader inputStream = null;
        Scanner scanned_line = null;
        String line;
        String[] value;
        value = new String[7];
        int countv;
        try {
            inputStream = new BufferedReader(new
            FileReader("debito-populate-table.txt"));
            while ((line = inputStream.readLine()) != null) {
                countv=0;

                System.out.println("<<");
                //split fields separated by tab delimiters
                scanned_line = new Scanner(line);
                scanned_line.useDelimiter("\t");

                while (scanned_line.hasNext()) {
                    System.out.println(value[countv++]=scanned_li
ne.next());
                } //while

                if (scanned_line != null) {
                    scanned_line.close();
                }

                System.out.println(">>");
                System.out.println("insert into debito (numero_debito,
                valor_debito, motivo_debito, data_debito,
                numero_conta, nome_agencia, nome_cliente) " +
                "values (" + value[0] + ", " + value[1] + ", " + value[2] + ",
                ""+ value[3] + "", " + value[4] + ", ""+ value[5] + "", ""+
                value[6] + "");" );

            } //while
        } finally { if (inputStream != null) { inputStream.close(); } } //if &
        finally
    } //main
} //class
```

```
ReadFile.java
~/Documentos/mydir/JDBCTutorial

1 import java.io.FileReader;
2 import java.io.BufferedReader;
3 import java.io.IOException;
4 import java.util.Scanner;
5 public class ReadFile {
6     public static void main(String[] args) throws IOException {
7         BufferedReader inputStream = null;
8         Scanner scanned_line = null;
9         String line;
10        String[] value;
11        value = new String[7];
12        int countv;
13        try {
14            inputStream = new BufferedReader(new FileReader("debito-populate-table.txt"));
15            while ((line = inputStream.readLine()) != null) {
16                countv=0;
17                System.out.println("<<");
18                //split fields separated by tab delimiters
19                scanned_line = new Scanner(line);
20                scanned_line.useDelimiter("\t");
21                while (scanned_line.hasNext()) {
22                    System.out.println(value[countv++]=scanned_line.next());
23                } //while
24
25                if (scanned_line != null) {
26                    scanned_line.close();
27                }
28
29                System.out.println(">>");
30                System.out.println("insert into debito (numero_debito, valor_debito,
31                motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values
32                (" + value[0] + ", " + value[1] + ", " + value[2] + ", " + value[3] + ", " + value[4] + ", " + value[5] + ", " +
33                value[6] + " );");
34            } //while
35        } finally {
36            if (inputStream != null) {
37                inputStream.close();
38            }
39        } //if & finally
40    }
41}
```

```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial
Arquivo Editar Ver Pesquisar Terminal Ajuda

euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ ls
BD2-14-postgresql-42.2.4.jar  comp.save.1  properties      txt
build.xml                    debito-populate-table.txt  ReadFile.java  xml
classes                      derby.log    sql
comp                         JDBCUtilities.java  src
comp.save                   lib          testdb
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$
```

6) Compile e execute o arquivo ReadFile.java:

1. javac ReadFile.java
2. java ReadFile

```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial
Arquivo Editar Ver Pesquisar Terminal Ajuda

euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ javac ReadFile.java
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ java ReadFile
<<
1001
900.75
4
2011-12-24
759
Pampulha
Pedro Alvares Sousa
>>
insert into debito (numero_debito, valor_debito,
motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (10
01, 900.75, 4, '2011-12-24', 759, 'Pampulha', 'Pedro Alvares Sousa');
<<
1002
858.81
5
2012-11-18
34526
Central
Marcos Andrade
>>
insert into debito (numero_debito, valor_debito,
```

7) Utilize a interface do pgadmin3 para criar a seguinte tabela no banco

Nome da tabela: debito

Nome da Coluna Tipo Modificadores

numero_debito integer not null

valor_debito double precision not null

motivo_debito smallint

data_debito date

numero_conta integer

nome_agencia character varying(50)

nome_cliente character varying(80)

Índices:

Nome: pk_debito, Tipo: PRIMARY KEY, Sobre: (numero_debito)

Restrições de chave estrangeira:

Nome: fk_debito, Tipo: FOREIGN KEY Sobre: (numero_conta, nome_agencia, nome_cliente),

Referenciando: conta(numero_conta, nome_agencia, nome_cliente)

Perceba que ao final temos apenas uma tabela sem dado algum e será necessário realizar um carga de dados na tabela

The image displays three screenshots of the pgAdmin 3 'New Table' dialog box, illustrating the configuration for a table named 'debito'.

Top Left Screenshot (Properties tab): Shows the 'Name' field set to 'debito', 'Schema' set to 'public', and 'Use Slony' set to 'No'.

Top Right Screenshot (Columns tab): Shows the column definitions for the 'debito' table:

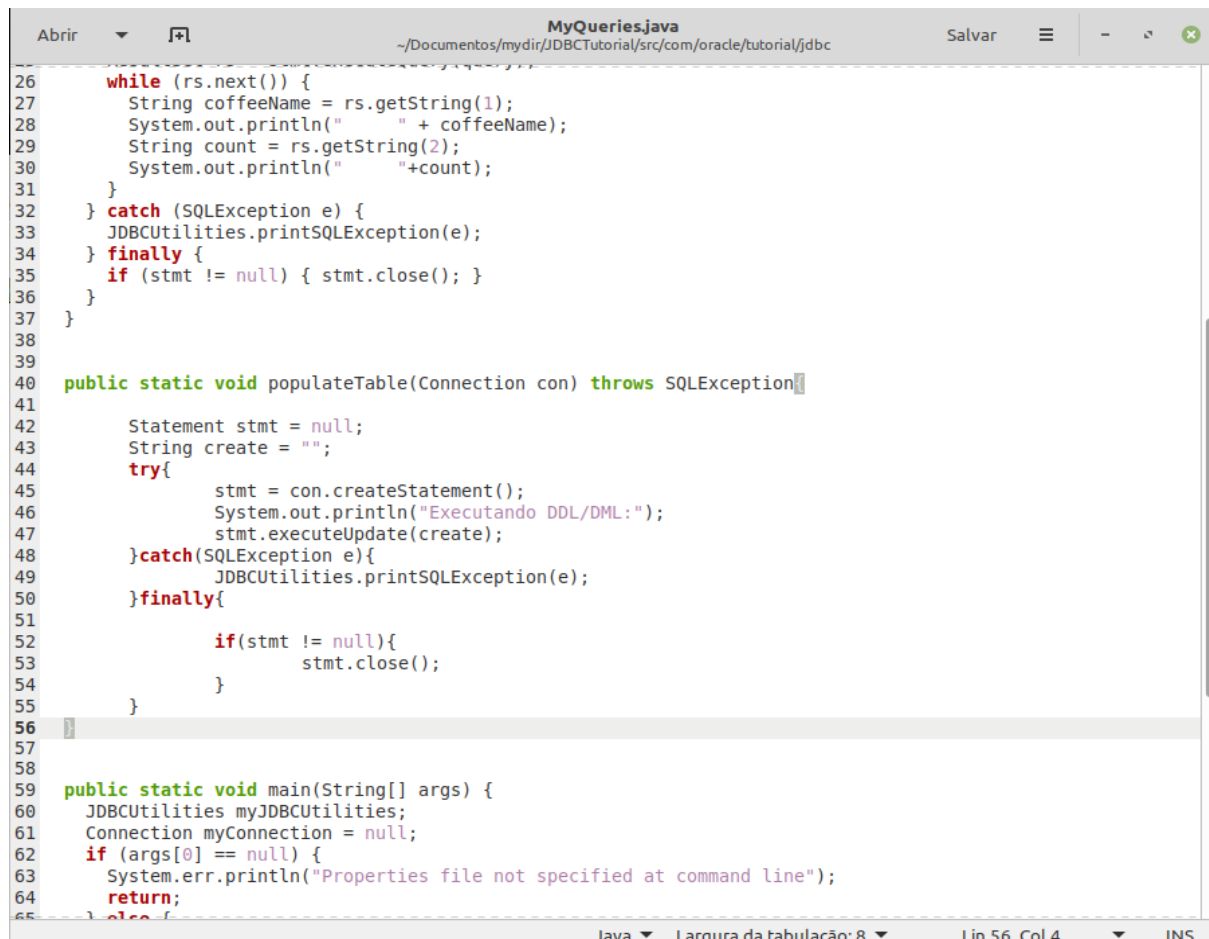
Column name	Definition	Inherited from table
numero_debito	integer NOT NULL	
valor_debito	double precision NOT NULL	
motivo_debito	smallint	
data_debito	date	
numero_conta	integer	
nome_agencia	character varying(50)	
nome_cliente	character varying(80)	

Bottom Screenshot (Constraints tab): Shows the constraints defined for the 'debito' table:

Constraint name	Definition
pk_debito	(numero_debito)
fk_debito	(numero_conta, nome_agencia, nome_cliente) REFERENCES public.conta (numero_conta, nome_agencia, nome_cliente) ON UPDATE NO ACTION ON DELETE NO ACTION

8) Após ter criado manualmente a tabela, agora é necessário populá-la com os dados do arquivo debito-populate.txt. Crie um método na classe MyQueries para executar tal operação. Utilize o arcabouço do código abaixo para este propósito

```
public static void populateTable(Connection con) throws SQLException {
    Statement stmt = null;
    String create = "";
    try {
        stmt = con.createStatement();
        System.out.println("Executando DDL/DML:");
        stmt.executeUpdate(create);
    } catch (SQLException e) {
        JDBCUtilities.printSQLException(e);
    } finally {
        if (stmt != null) { stmt.close(); }
    }
}
```



```
Abriu  MyQueries.java  Salvar
~/Documentos/mydir/JDBCTutorial/src/com/oracle/tutorial/jdbc

26 while (rs.next()) {
27     String coffeeName = rs.getString(1);
28     System.out.println("    " + coffeeName);
29     String count = rs.getString(2);
30     System.out.println("    "+count);
31 }
32 } catch (SQLException e) {
33     JDBCUtilities.printSQLException(e);
34 } finally {
35     if (stmt != null) { stmt.close(); }
36 }
37 }
38
39
40 public static void populateTable(Connection con) throws SQLException
41
42     Statement stmt = null;
43     String create = "";
44     try{
45         stmt = con.createStatement();
46         System.out.println("Executando DDL/DML:");
47         stmt.executeUpdate(create);
48     }catch(SQLException e){
49         JDBCUtilities.printSQLException(e);
50     }finally{
51
52         if(stmt != null){
53             stmt.close();
54         }
55     }
56
57
58
59 public static void main(String[] args) {
60     JDBCUtilities myJDBCUtilities;
61     Connection myConnection = null;
62     if (args[0] == null) {
63         System.err.println("Properties file not specified at command line");
64         return;
65     } else {
```

Perceba que o código aqui tem um comando de execução diferente dos anteriores. A explicação está abaixo na Figura 1, em resumo: o método executeQuery() retorna um objeto do tipo ResultSet, mas nem todos os comandos executados no SGBD vão retornar algum valor a ser tratado, como é o caso dos comandos DDL e DML. Para comandos que não retornam parâmetros utilizamos o método executeUpdate().

9) O código ReadFile.java deve ser reaproveitado para popular a tabela 'debito'. Para tanto:

1. Mescle-o com o código apresentado na classe ReadFile.java;

1. As cláusulas import devem ser inseridas no início do MyQueries.java;

```
1 package com.oracle.tutorial.jdbc;
2
3 import java.sql.Connection;
4 import java.sql.ResultSet;
5 import java.sql.SQLException;
6 import java.sql.Statement;
7 import java.io.FileReader;
8 import java.io.BufferedReader;
9 import java.io.IOException;
10 import java.util.Scanner;
```

2. O código destacado em negrito no ReadFile.java não fará parte do método populatetable();

3. O código com tipo de letra normal (sem negrito) deve ser inserido dentro da cláusula try { ... }, antes do acionamento do comando “stmt.executeUpdate(create)”. Neste caso, ao invés da impressão do resultado na tela, devemos armazenar o resultado da cadeia de caracteres criada (comando insert into) dentro da variável create.

```
public static void populateTable(Connection con) throws SQLException, IOException{
    Statement stmt = null;
    BufferedReader inputStream = null;
    Scanner scanned_line = null;
    String line;
    String[] value;
    String create;
    value = new String[7];
    int countv;

    try{
        stmt = con.createStatement();
        inputStream = new BufferedReader(new FileReader("/home/euller/Documents/mydir/JDBCTutorial/debito-populate-table.txt"));

        while ((line = inputStream.readLine()) != null) {

            create = "";
            countv=0;

            scanned_line = new Scanner(line);
            scanned_line.useDelimiter("\t");
            while (scanned_line.hasNext()) {
                value[countv++]=scanned_line.next();
            } //while

            if (scanned_line != null) {
                scanned_line.close();
            }

            create = "insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) " + "values (" +
            value[0] + ", " + value[1] + ", " + value[2] + ", " + value[3] + ", " + value[4] + ", " + value[5] + ", " + value[6] + " )";

            System.out.println("\nExecutando DDL/DML:");
            System.out.println(create+"\n");
            stmt.executeUpdate(create);

        } //while
    } catch (SQLException e){
        JDBCUtilities.printSQLException(e);
    } catch (IOException e){
        //
    }
}
```


4. Atenção especial deve ser dada para o tratamento de erros. Antes o código `createtable()`, agora transformado em `populateTable()` precisava se resguardar da possibilidade de disparar (throws) mensagens de erro de comandos em SQL, por isso a classe tinha a cláusula `throws SQLException` na definição do método e ao final da cláusula `try { ... }` havia um “`catch (SQLException e)`”, estrutura trabalha na última aula. Como o código incorporado possui a possibilidade de que ocorram erros de manipulação de arquivos, precisamos assegurar que a cláusula `try { ... }` poderá ter a chance de tratar erros disparados na manipulação de arquivos. Para tanto, acrescente:

1. “`throws SQLException, IOException`”, na definição do método;

```
public static void populateTable(Connection con) throws SQLException, IOException{
```

2. “`catch (IOException e) { e.printStackTrace(); }`” após o `catch` existente;

```
    }catch(SQLException e){
        JDBCUtilities.printSQLException(e);
    }catch(IOException e){
        e.printStackTrace();
    }
    finally{
        if(stmt != null){
            stmt.close();
        }
    }
}
```

5. Coloque antes do nome do arquivo `debito-populate-table.txt` o caminho completo de sua localização no disco;

```
inputStream = new BufferedReader(new FileReader("/home/euller/Documents/mydir/JDBCTutorial/debito-populate-table.txt"));
```

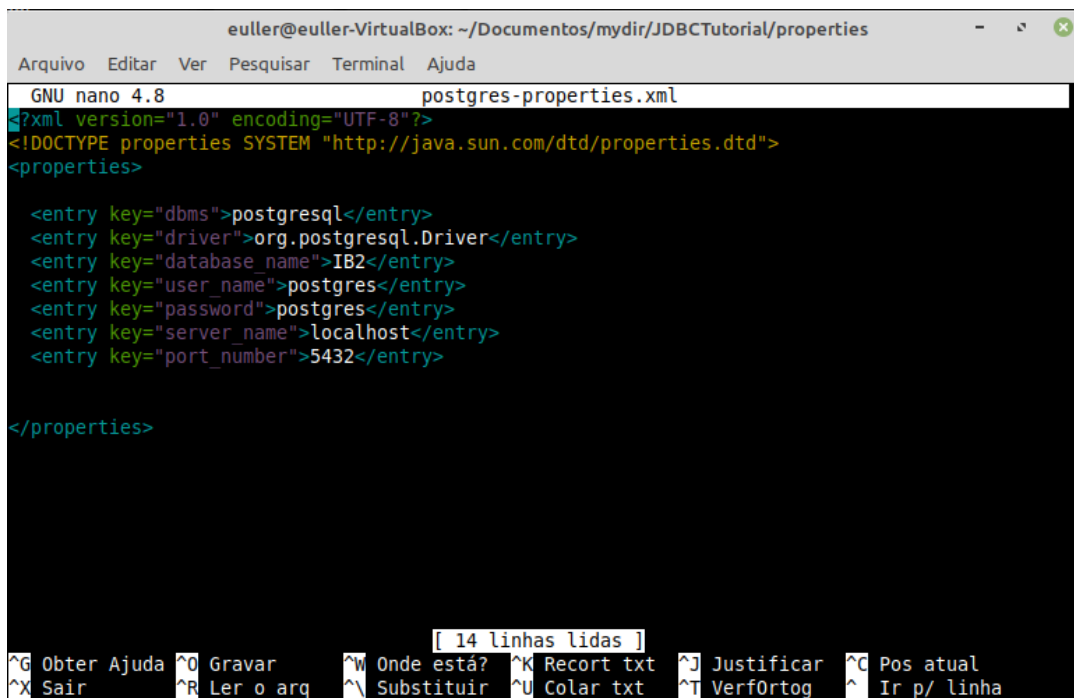
2. Inclua a chamada ao método `populateTable()` no método `main` da classe `MyQueries`;

```
public static void main(String[] args) {
    JDBCUtilities myJDBCUtilities;
    Connection myConnection = null;
    if (args[0] == null) {
        System.err.println("Properties file not specified at command line");
        return;
    } else {
        try {
            myJDBCUtilities = new JDBCUtilities(args[0]);
        } catch (Exception e) {
            System.err.println("Problem reading properties file " + args[0]);
            e.printStackTrace();
            return;
        }
    }

    try {
        myConnection = myJDBCUtilities.getConnection();
        MyQueries.populateTable(myConnection);

    } catch (SQLException e) {
        JDBCUtilities.printSQLException(e);
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        JDBCUtilities.closeConnection(myConnection);
    }
}
```

10) Modificar/criar o arquivo de propriedades de conexão para acessar o banco IB2. Recomendo que copie o arquivo de propriedades de conexão anterior ao PostgreSQL para fazer a conexão ao banco IB2



```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial/properties
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
GNU nano 4.8 postgres-properties.xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>

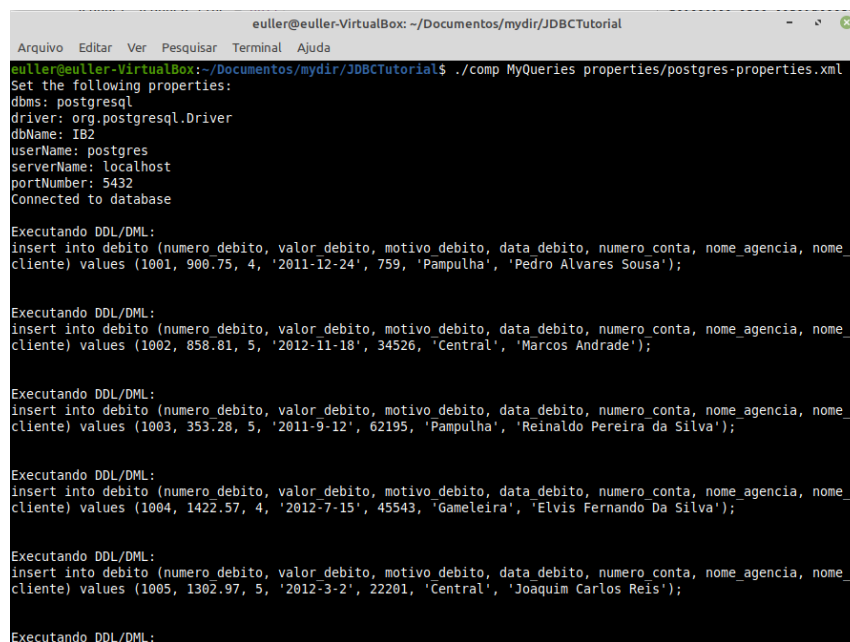
  <entry key="dbms">postgresql</entry>
  <entry key="driver">org.postgresql.Driver</entry>
  <entry key="database_name">IB2</entry>
  <entry key="user_name">postgres</entry>
  <entry key="password">postgres</entry>
  <entry key="server_name">localhost</entry>
  <entry key="port_number">5432</entry>

</properties>

[ 14 linhas lidas ]
^G Obter Ajuda ^O Gravar ^W Onde está? ^K Recort txt ^J Justificar ^C Pos atual
^X Sair ^R Ler o arq ^_ Substituir ^U Colar txt ^T Verf0rtog ^_ Ir p/ linha
```

11) Salve, compile e execute o MyQueries.

1. Certifique-se de que o arquivo properties/postgres-properties.xml está configurado com os parâmetros de conexão ao banco de dados correto;
2. ./comp MyQueries properties/postgres-properties.xml



```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ ./comp MyQueries properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to database

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1001, 900.75, 4, '2011-12-24', 759, 'Pampulha', 'Pedro Alvares Sousa');

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1002, 858.81, 5, '2012-11-18', 34526, 'Central', 'Marcos Andrade');

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1003, 353.28, 5, '2011-9-12', 62195, 'Pampulha', 'Reinaldo Pereira da Silva');

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1004, 1422.57, 4, '2012-7-15', 45543, 'Gameleira', 'Elvis Fernando Da Silva');

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1005, 1302.97, 5, '2012-3-2', 22201, 'Central', 'Joaquim Carlos Reis');

Executando DDL/DML:
```

12) Sempre que você tentar executar a inserção destes dados mais de uma vez na tabela debito serão exibidas mensagens de erro como esta:

org.postgresql.util.PSQLException: ERROR: duplicate key value violates unique constraint "pk_debito"

Execute o código e comprove.

./comp MyQueries properties/postgres-properties.xml

```
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ ./comp MyQueries properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to database

Executando DDL/DML:
insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) values (1001, 900.75, 4, '2011-12-24', 759, 'Pampulha', 'Pedro Alvares Sousa');
org.postgresql.util.PSQLException: ERROR: duplicate key value violates unique constraint "pk_debito"
  Detalhe: Key (numero debito)=(1001) already exists.
    at org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:2440)
    at org.postgresql.core.v3.QueryExecutorImpl.processResults(QueryExecutorImpl.java:2183)
    at org.postgresql.core.v3.QueryExecutorImpl.execute(QueryExecutorImpl.java:308)
    at org.postgresql.jdbc.PgStatement.executeInternal(PgStatement.java:441)
    at org.postgresql.jdbc.PgStatement.execute(PgStatement.java:365)
    at org.postgresql.jdbc.PgStatement.executeWithFlags(PgStatement.java:307)
    at org.postgresql.jdbc.PgStatement.executeCachedSql(PgStatement.java:293)
    at org.postgresql.jdbc.PgStatement.executeWithFlags(PgStatement.java:270)
    at org.postgresql.jdbc.PgStatement.executeUpdate(PgStatement.java:244)
    at com.oracle.tutorial.jdbc.MyQueries.populateTable(MyQueries.java:79)
    at com.oracle.tutorial.jdbc.MyQueries.main(MyQueries.java:116)
SQLState: 23505
Error Code: 0
Message: ERROR: duplicate key value violates unique constraint "pk_debito"
  Detalhe: Key (numero debito)=(1001) already exists.
Releasing all open resources ...
```

Isso ocorre porque a integridade da chave primária é policiada pelo SGBD. Para evitar que esta mensagem de erro ocorra, supondo que a tabela precisa ser recarregada periodicamente, podemos usar um comando para truncar a tabela antes de novas inserções. Insira o comando antes do loop de inserção de linhas lidas do arquivo, salve e execute novamente:

1. stmt.executeUpdate("truncate table debito;");

```
public static void populateTable(Connection con) throws SQLException, IOException{
    Statement stmt = null;
    BufferedReader inputStream = null;
    Scanner scanned_line = null;
    String line;
    String[] value;
    String create;
    value = new String[7];
    int countv;

    try{
        stmt = con.createStatement();
        inputStream = new BufferedReader(new FileReader("/home/euller/Documentos/mydir/JDBCTutorial/debito-populate-table.txt"));
        stmt.executeUpdate("truncate table debito");
        while ((line = inputStream.readLine()) != null) {

            create = "";
            countv=0;

            scanned_line = new Scanner(line);
            scanned_line.useDelimiter("\t");
            while (scanned_line.hasNext()) {
                value[countv++]=scanned_line.next();
            } //while

            if (scanned_line != null) {
                scanned_line.close();
            }

            create = "insert into debito (numero debito, valor debito, motivo debito, data debito, numero conta, nome_agencia, nome_cliente) " + "values (" +
            alue[0] + ", " + value[1] + ", " + value[2] + ", " + value[3] + ", " + value[4] + ", " + value[5] + ", " + value[6] + ")";

            System.out.println("\nExecutando DDL/DML:");
            System.out.println(create+"\n");
            stmt.executeUpdate(create);

        } //while
    }
```

2. ./comp MyQueries properties/postgres-properties.xml

```
euller@euller-VirtualBox: ~/Documentos/mydir/JDBCTutorial
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
euller@euller-VirtualBox:~/Documentos/mydir/JDBCTutorial$ ./comp MyQueries properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to database

Executando DDL/DML:
insert into debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (1001, 900.75, 4, '2011-12-24', 759, 'Pampulha', 'Pedro Alvares Sousa');

Executando DDL/DML:
insert into debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (1002, 858.81, 5, '2012-11-18', 34526, 'Central', 'Marcos Andrade');

Executando DDL/DML:
insert into debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (1003, 353.28, 5, '2011-9-12', 62195, 'Pampulha', 'Reinaldo Pereira da Silva');

Executando DDL/DML:
insert into debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (1004, 1422.57, 4, '2012-7-15', 45543, 'Gameleira', 'Elvis Fernando Da Silva');

Executando DDL/DML:
insert into debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) values (1005, 1302.97, 5, '2012-3-2', 22201, 'Central', 'Joaquim Carlos Reis');

Executando DDL/DML:
```