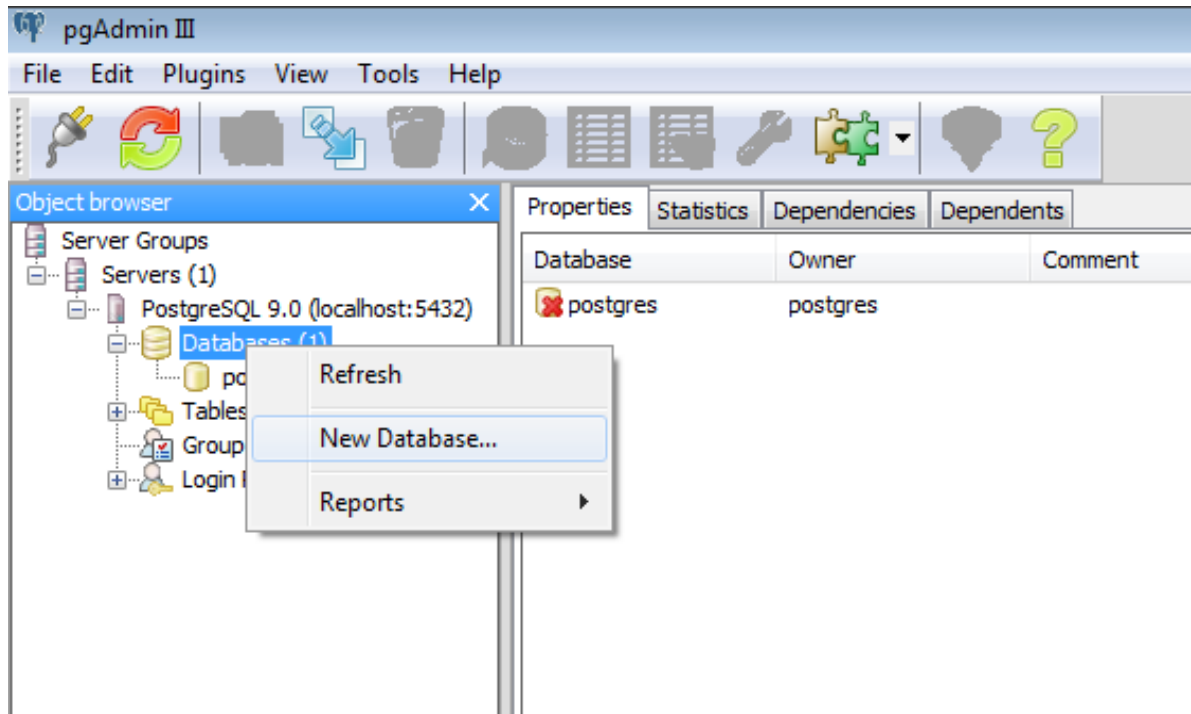


Manipulação de Banco de Dados com Java/JDBC

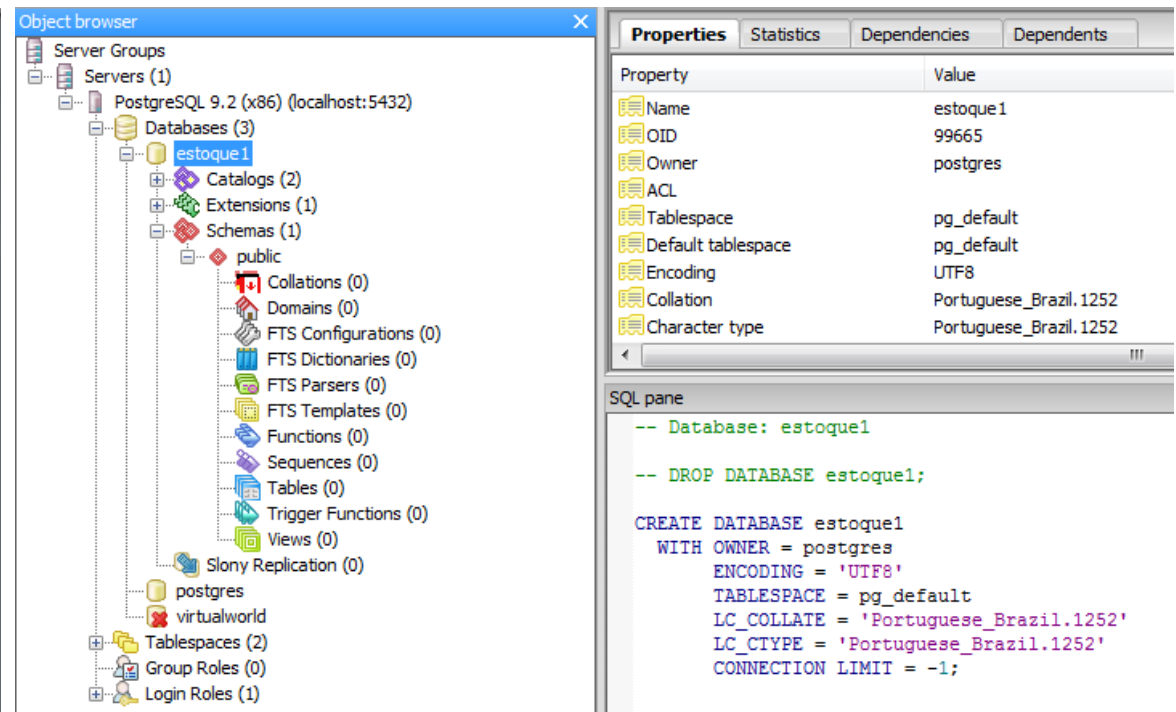
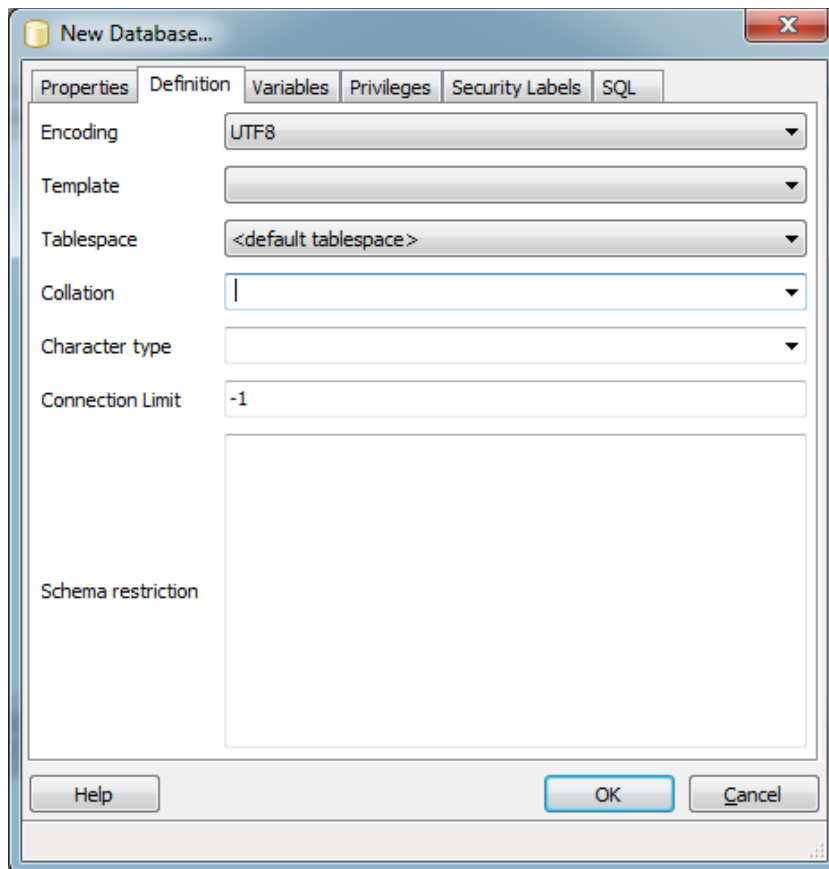
SGDB: PostgreSQL

Nome do Banco de Dados (Database): [estoque1](#)

Usando o pgAdmin III para criar a nova base de dados (botão direito do mouse em Databases/New Database...



Definição dos parâmetros da nova base de dados



Criar as Tabelas

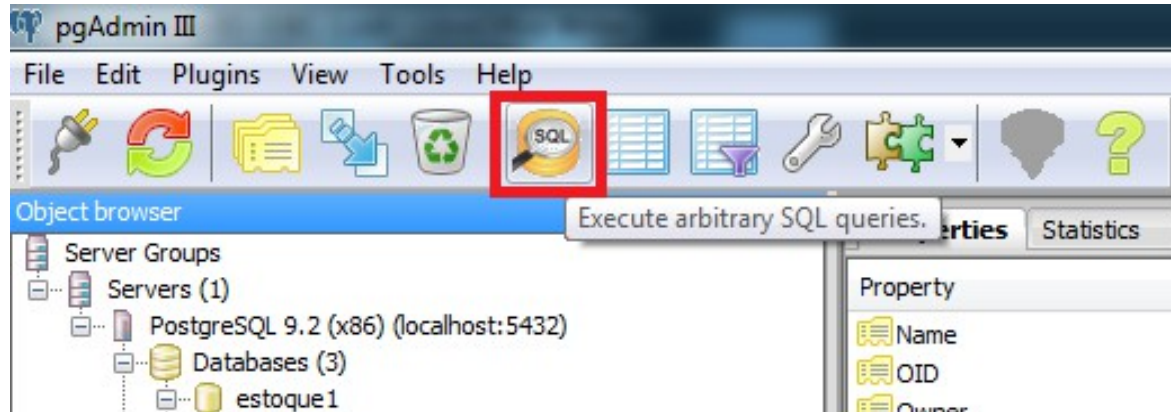


Tabela – grupoproduto

```
CREATE TABLE grupoproduto (  
  codigo serial NOT NULL,  
  nome character(40) NOT NULL,  
  promocao numeric(5,2),  
  margemlucro numeric(5,2),  
  PRIMARY KEY (codigo)  
)
```

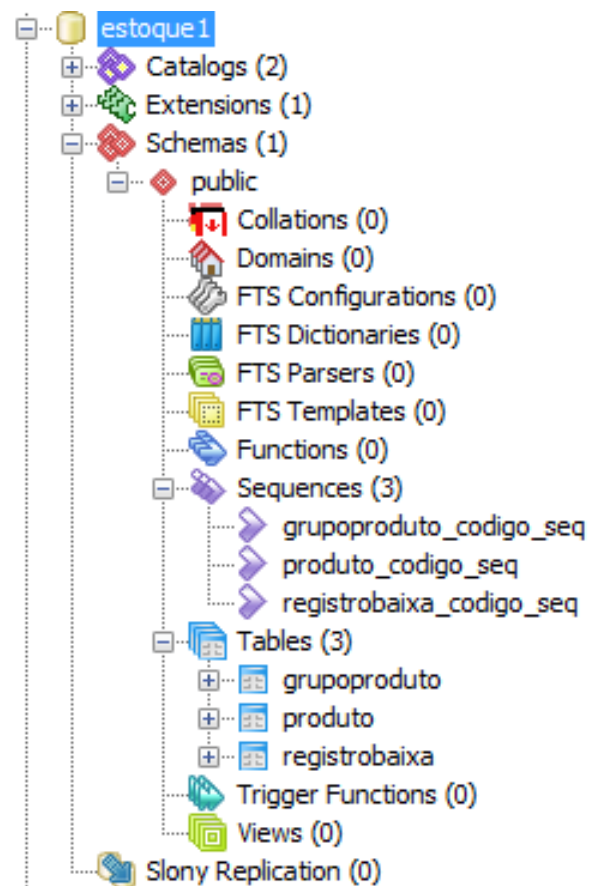
Tabela – produto

```
CREATE TABLE produto (  
  codigo SERIAL NOT NULL,  
  nome CHARACTER(50) NOT NULL,  
  estoque INTEGER,  
  valorcompra NUMERIC(10,2),  
  promocao NUMERIC(5,2),  
  margemlucro NUMERIC(5,2),  
  grupo INTEGER NOT NULL,  
  PRIMARY KEY (codigo),  
  FOREIGN KEY (grupo) REFERENCES grupoproduto (codigo)  
)
```

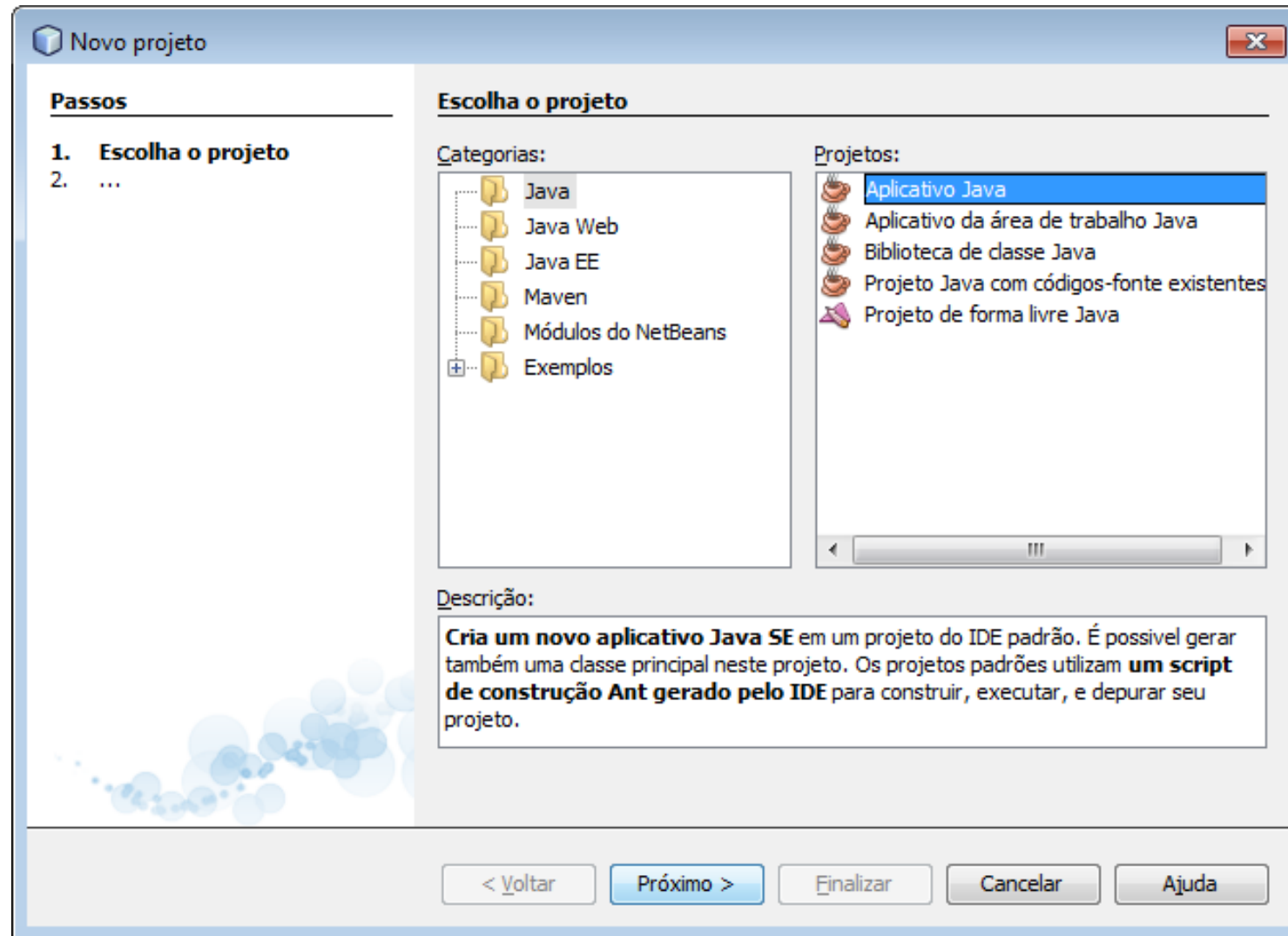
Tabela – registrobaixa

```
CREATE TABLE registrobaixa (  
  codigo SERIAL NOT NULL,  
  motivo SMALLINT NOT NULL,  
  data DATE NOT NULL,  
  quantidade INTEGER NOT NULL,  
  produto INTEGER NOT NULL,  
  PRIMARY KEY (codigo),  
  FOREIGN KEY (produto) REFERENCES produto (codigo)  
)
```

Base de dados **estoque1** após execução dos *scripts*



Criar um Projeto no Netbeans para teste de conexão e manipulação da base de dados [estoque1](#).



Continuação...

Novo Aplicativo Java

Passos

1. Escolha o projeto
2. **Nome e local**

Nome e local

Nome do projeto: ManipulacaoBD

Localização do Projeto: D:\NetBeansProjects **Procurar...**

Pasta do projeto: D:\NetBeansProjects\ManipulacaoBD

☐ Usar pasta dedicada para armazenar bibliotecas

Pasta Bibliotecas: **Procurar...**

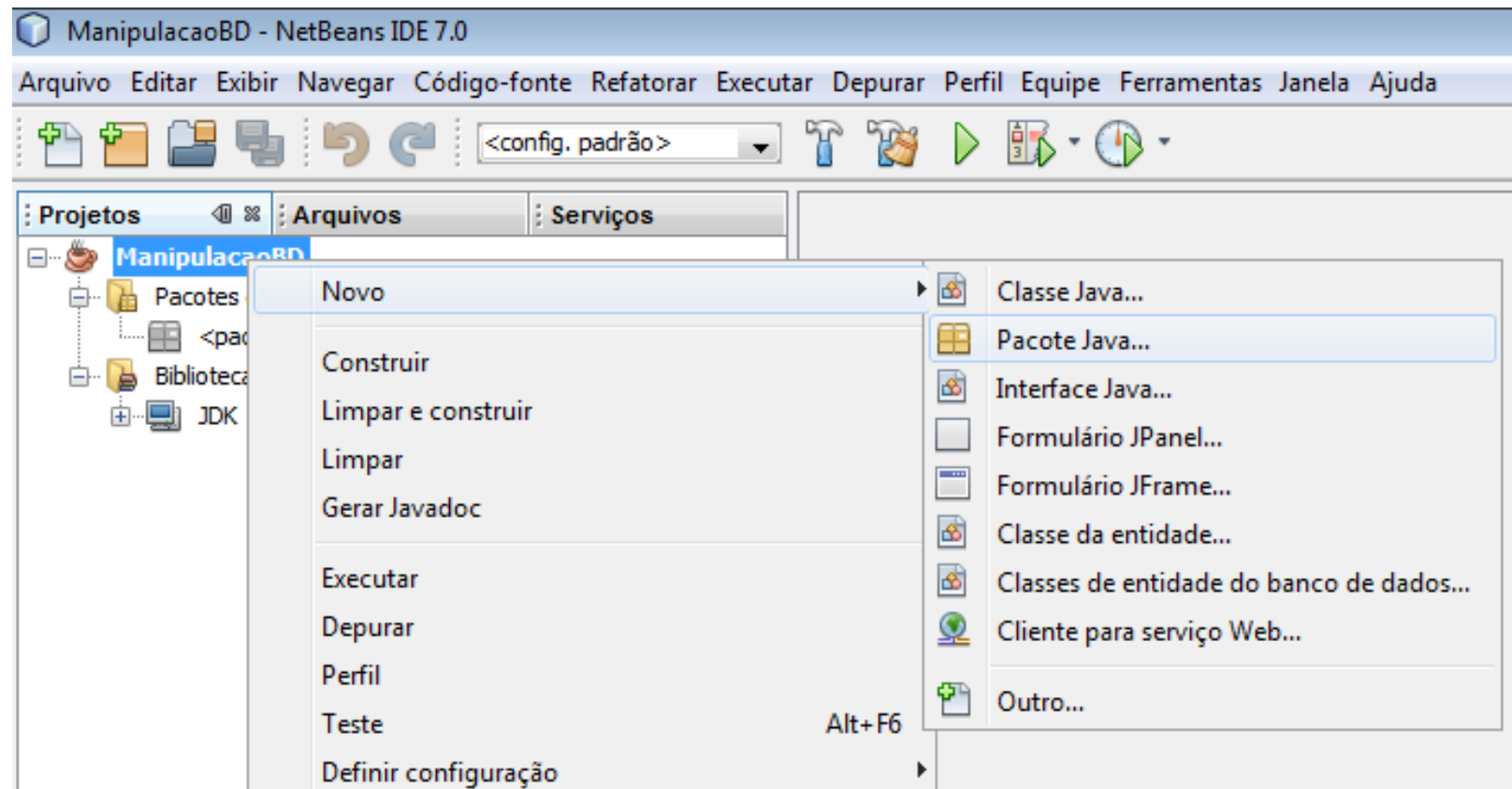
Usuários e projetos diferentes podem compartilhar as mesmas bibliotecas de compilação (consulte a Ajuda para obter detalhes).

☐ Criar classe principal

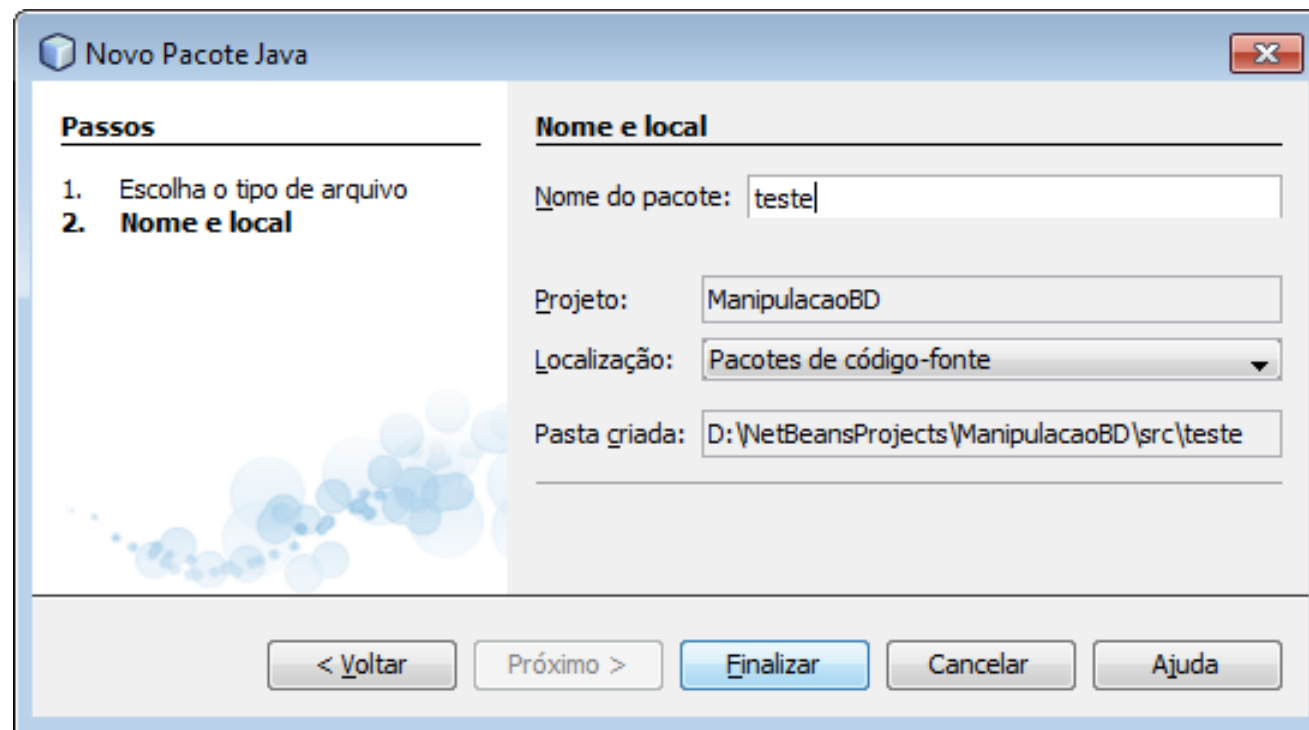
☒ Definir como projeto principal

< Voltar Próximo > **Finalizar** Cancelar Ajuda

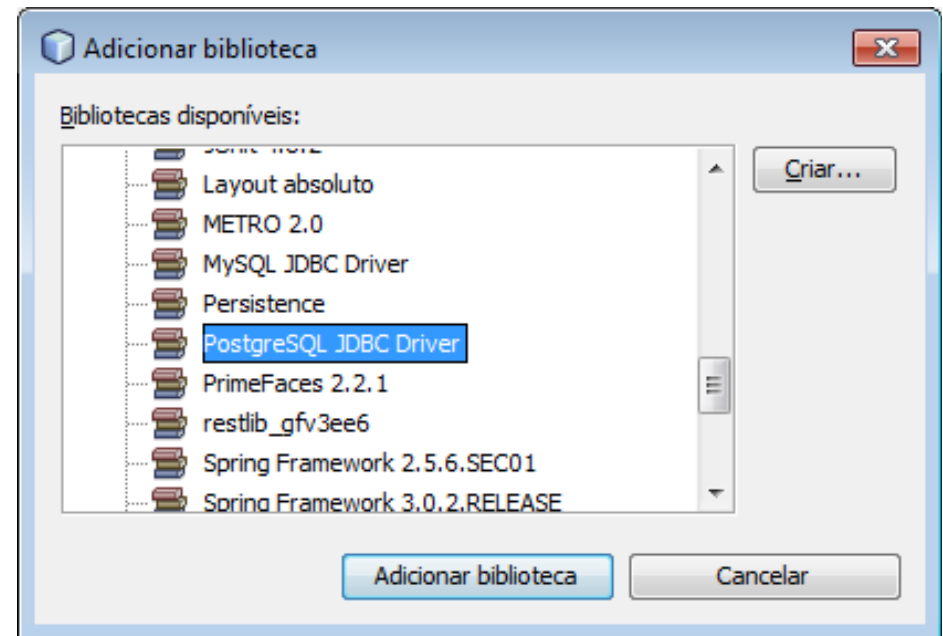
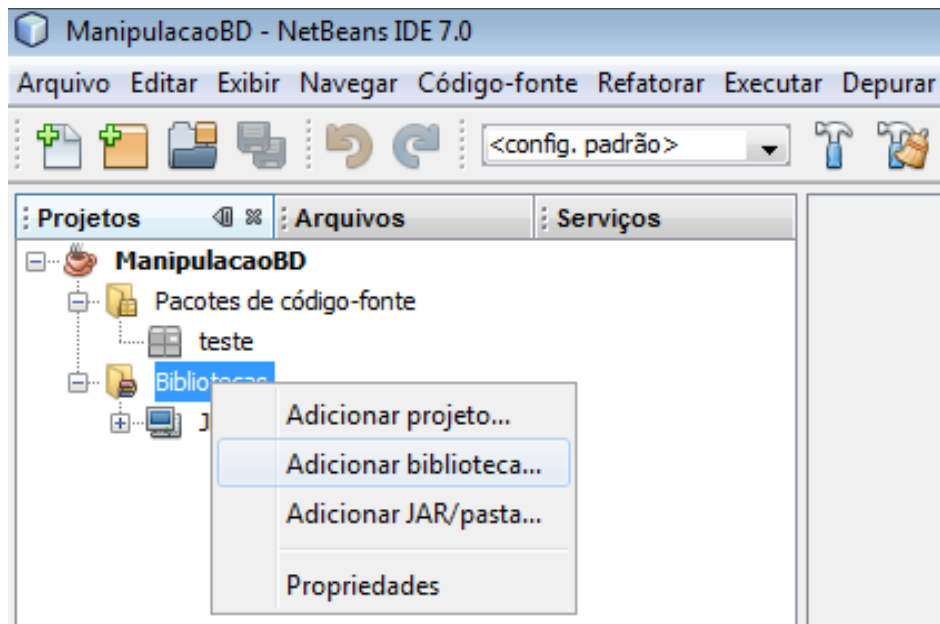
Criar um Pacote Java para conter as classes de testes. Botão direito no nome do projeto/Novo/Pacote Java...



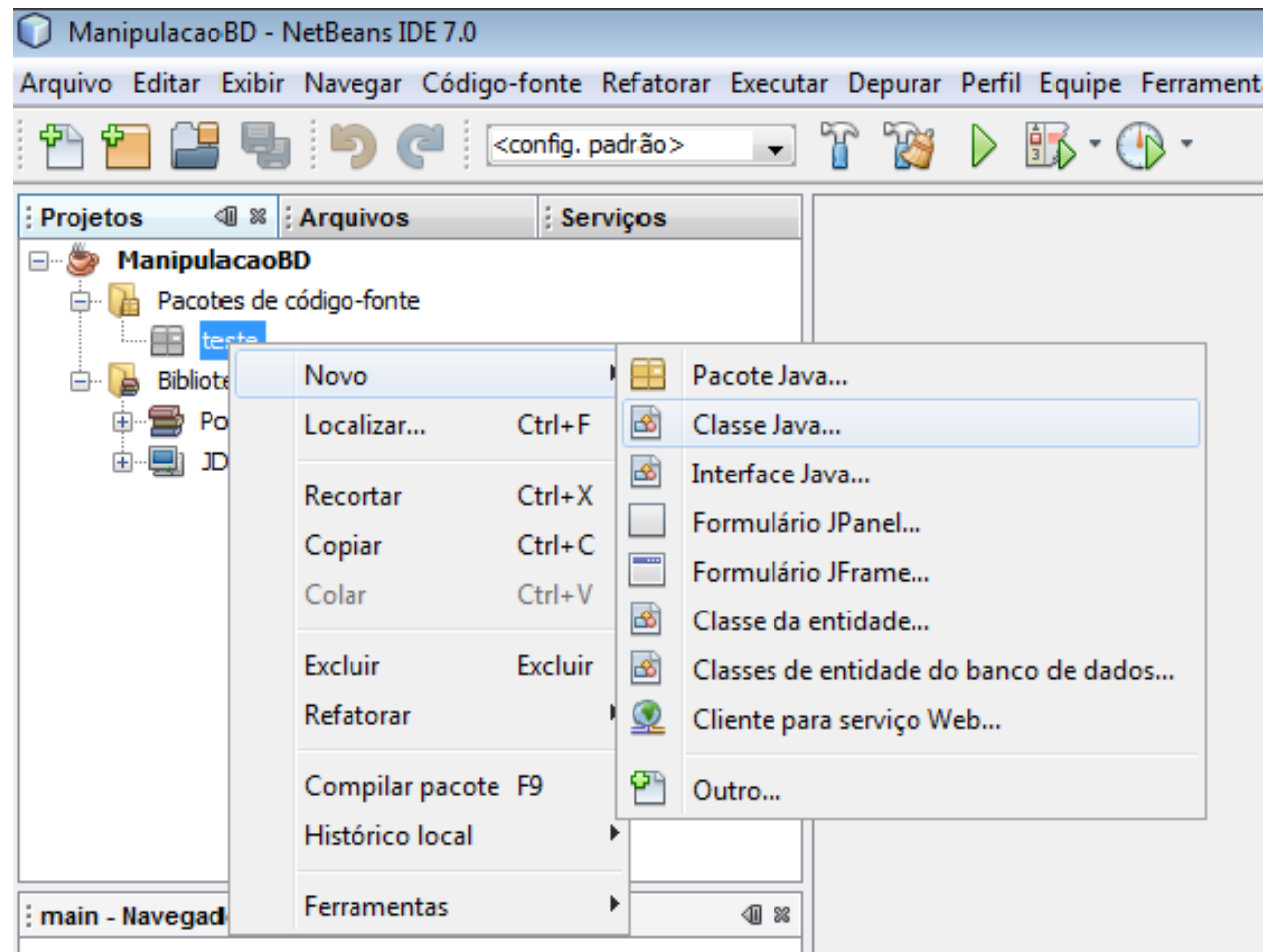
Defina o nome do pacote conforme imagem.



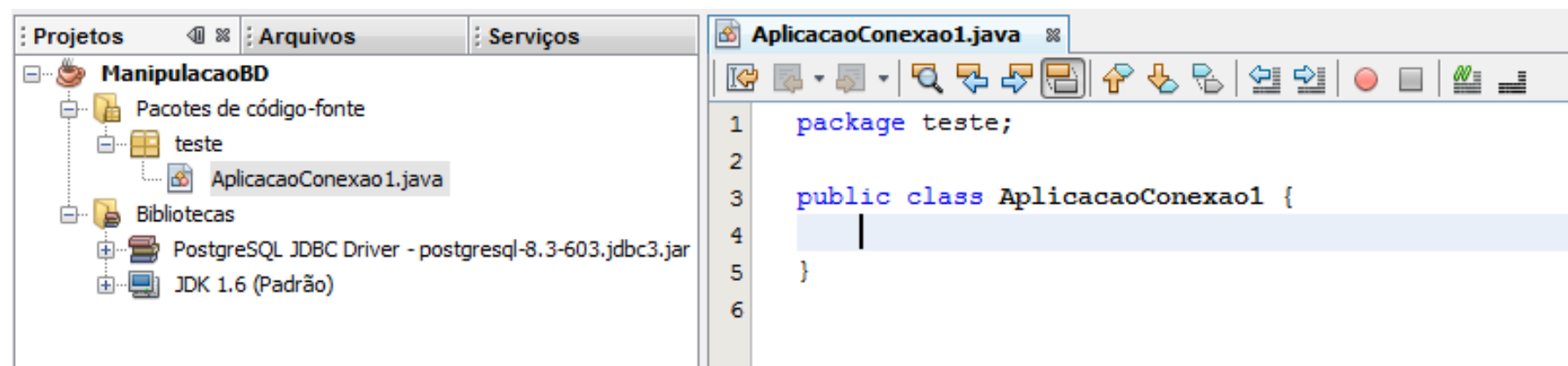
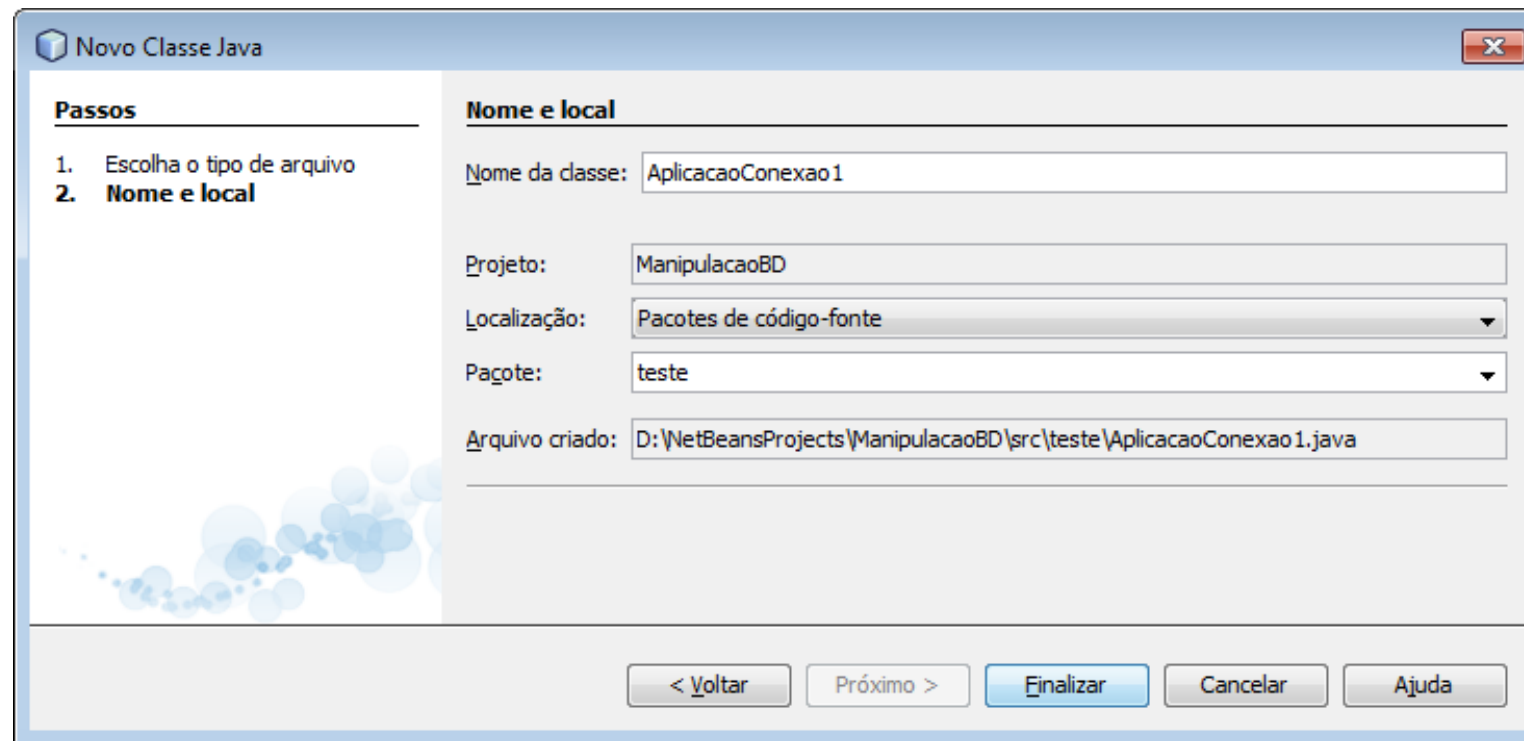
Adicionar o Driver JDBC/PostgreSQL no projeto. Botão direito do mouse sobre Bibliotecas/Adicionar Biblioteca...



Criar a primeira classe de teste de conexão. Botão direito do mouse sobre o pacote teste/Novo/Classe Java...



Forneça o nome da classe: AplicacaoConexao1



Exemplo01 – Conexão com o Banco de Dados – Class.forName para carga do driver do banco

```

1  package teste;
2  import java.sql.Connection;
3  import java.sql.DriverManager;
4  import java.sql.SQLException;
5  import javax.swing.JOptionPane;
6
7  public class AplicacaoConexao1 {
8      public static void main(String args[]){
9          Connection conexao;
10         String url = "jdbc:postgresql://localhost:5432/estoque1";
11         try{
12             Class.forName("org.postgresql.Driver");
13             conexao = DriverManager.getConnection(url,"postgres","postgres");
14             JOptionPane.showMessageDialog(null,"Conexao estabelecida");
15             conexao.close();
16         } catch(ClassNotFoundException cnf){
17             JOptionPane.showMessageDialog(null,"Driver nao encontrado - "+cnf.getMessage());
18         } catch(SQLException sqle){
19             JOptionPane.showMessageDialog(null,"Banco não conectado - "+sqle.getMessage());
20         }
21     }
22 }


```

Exemplo02 – Conexão com o Banco de Dados – DriverManager.registerDriver

```

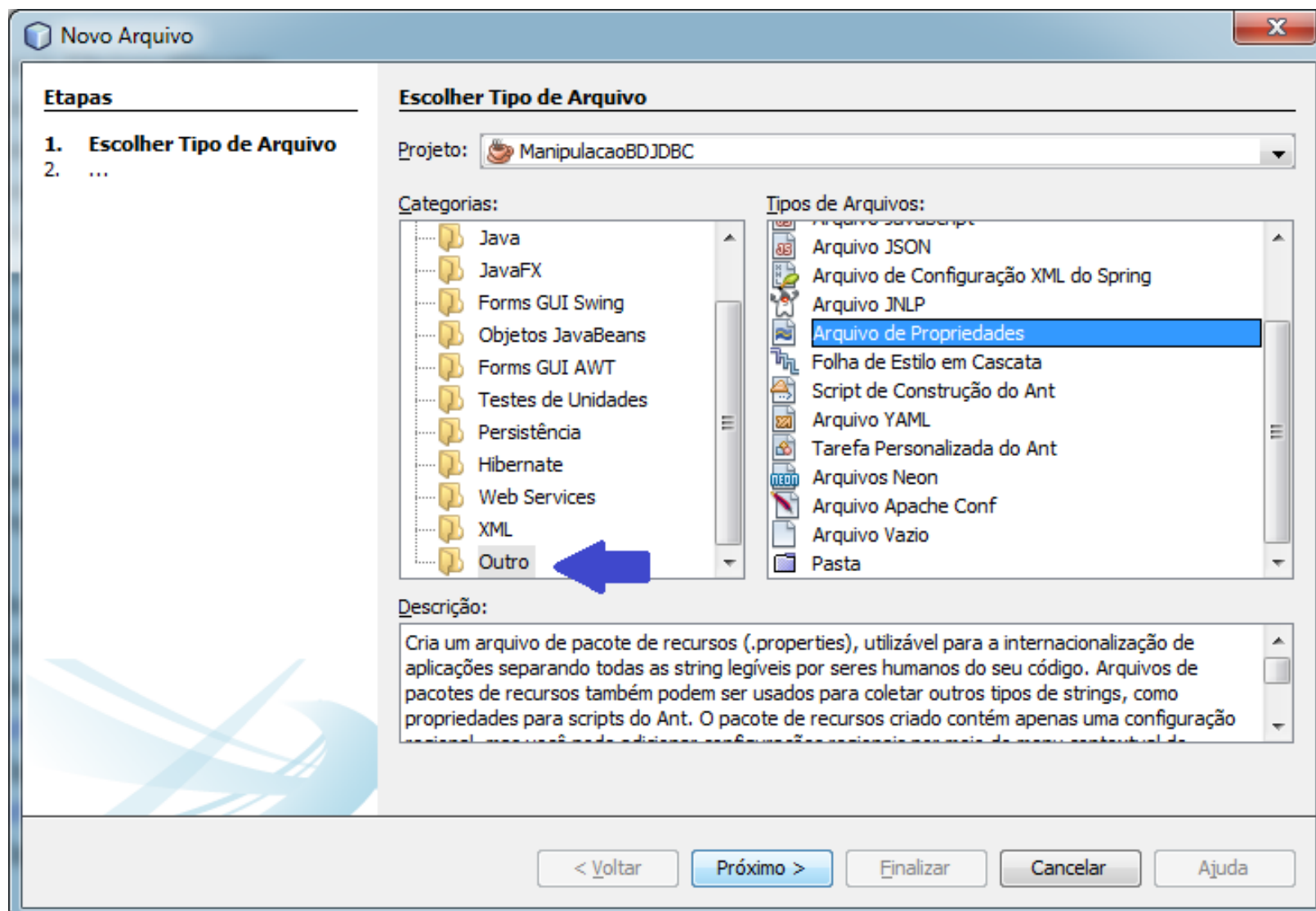
1  package teste;
2  import java.sql.Connection;
3  import java.sql.DriverManager;
4  import java.sql.SQLException;
5  import javax.swing.JOptionPane;
6  import org.postgresql.Driver;
7
8  public class AplicacaoConexao2 {
9      public static void main(String args[]){
10         Connection conexao;
11         String url = "jdbc:postgresql://localhost:5432/estoque1";
12         try{
13             DriverManager.registerDriver (new Driver());
14             conexao = DriverManager.getConnection(url,"postgres","postgres");
15             JOptionPane.showMessageDialog(null,"Conexao estabelecida");
16             conexao.close();
17         }catch(SQLException sqle){
18             JOptionPane.showMessageDialog(null,"Banco não conectado - "+sqle.getMessage());
19         }
20     }
21 }

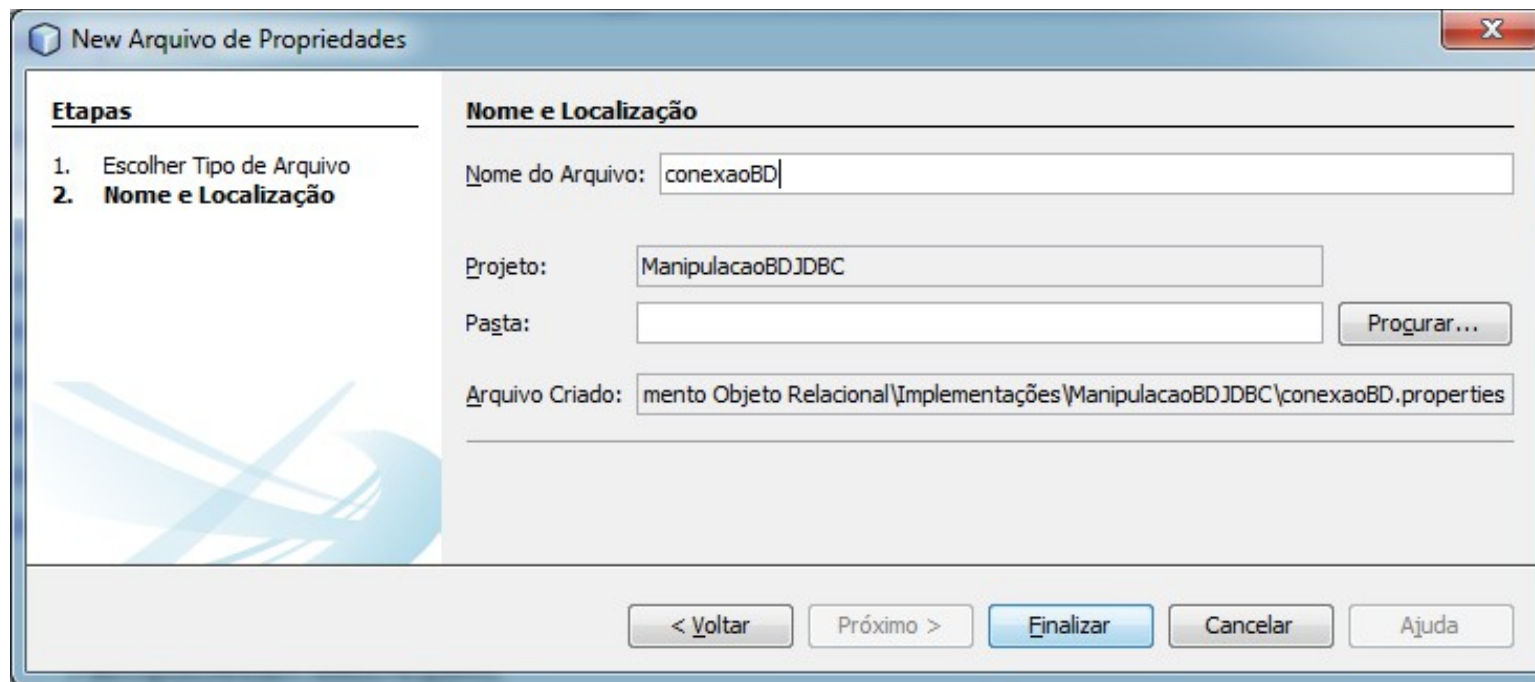
```



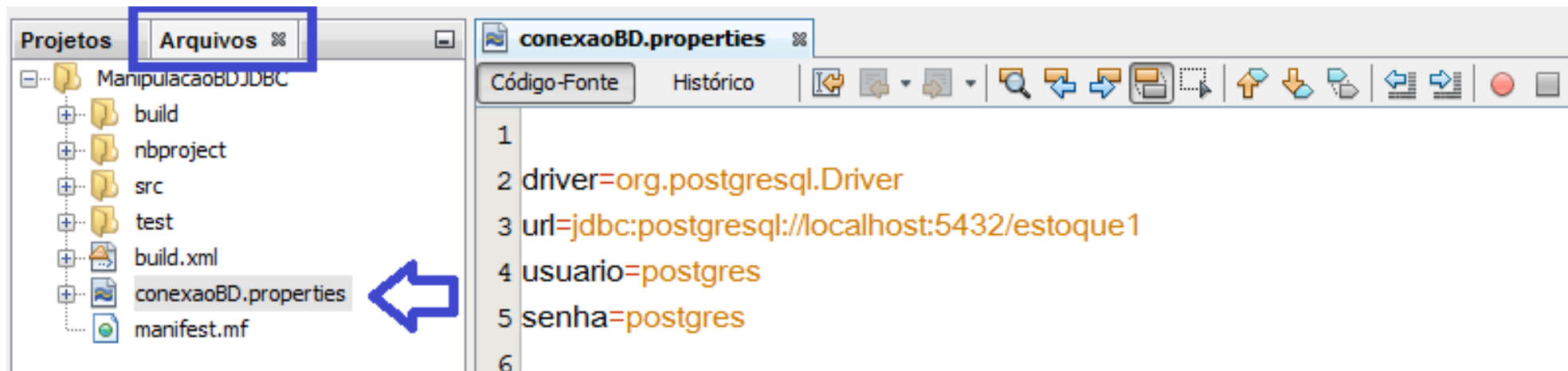
Exemplo03 – Conexão com o Banco de Dados – parâmetros em arquivo de propriedades

Botão direito no nome do projeto/Novo/Outros






Arquivo: conexaoBD.properties




Classe AplicacaoConexao3


```
1 package teste;
2 +import ...
10 public class AplicacaoConexao3 {
11 - public static void main(String args[]) {
12     Properties proBD = new Properties();
13     FileInputStream leitorArquivo;
14     try {
15         leitorArquivo = new FileInputStream("conexaoBD.properties");
16         proBD.load(leitorArquivo);
17         leitorArquivo.close();
18     } catch (FileNotFoundException ex) {
19         JOptionPane.showMessageDialog(null, "Configuracoes nao encontradas - " + ex.getMessage());
20     } catch (IOException ex) {
21         JOptionPane.showMessageDialog(null, "Erro ao ler configuracoes - " + ex.getMessage());
22     }
23
24     Connection conexao;
```



Classe AplicacaoConexao3 – Continuação...

```
25
26 String url = proBD.getProperty("url");
27 try {
28     Class.forName(proBD.getProperty("driver"));
29     conexao = DriverManager.getConnection(url, proBD.getProperty("usuario"), proBD.getProperty("senha"));
30     JOptionPane.showMessageDialog(null, "Conexao estabelecida");
31     conexao.close();
32 } catch (ClassNotFoundException cnf) {
33     JOptionPane.showMessageDialog(null, "Driver nao encontrado - " + cnf.getMessage());
34 } catch (SQLException sqle) {
35     JOptionPane.showMessageDialog(null, "Banco não conectado - " + sqle.getMessage());
36 }
37 }
38 }
```



Exemplo04 – Inclusão de Dados no Banco - Statement/createStatement

```

1  package teste;
2  import ...
7  public class AplicacaoIncluir1 {
8
9  private static Connection obterConexao(){
10      Connection conexao=null;
11      String url = "jdbc:postgresql://localhost:5432/estoque1";
12      try{
13          Class.forName("org.postgresql.Driver");
14          conexao = DriverManager.getConnection(url,"postgres","postgres");
15      } catch(ClassNotFoundException cnf){
16          JOptionPane.showMessageDialog(null,"Driver não encontrado - "+cnf.getMessage());
17      } catch(SQLException sqle){
18          JOptionPane.showMessageDialog(null,"Banco não conectado - "+sqle.getMessage());
19      }
20      return conexao;
21  }

```

Exemplo04 – Continuação...

```

23 public static void main(String args[]){
24     Connection conexao = obterConexao();
25     Statement comando = null;
26     String sql = "INSERT INTO GRUPOPRODUTO ( NOME, PROMOCAO, MARGEMLUCRO ) VALUES ";
27     String nome = "Bebidas destiladas";
28     float promocao = 10;
29     float margem = 40;
30     sql += "("+nome+", "+promocao+", "+margem+")";
31     try {
32         comando = conexao.createStatement();
33         comando.executeUpdate(sql);
34         JOptionPane.showMessageDialog(null, "Inclusão realizada com sucesso");
35     } catch (SQLException ex) {
36         JOptionPane.showMessageDialog(null, "Erro ao incluir grupo de produto" + ex.toString());
37     } finally{
38         try {
39             comando.close();
40             conexao.close();
41         } catch (SQLException ex) {
42             JOptionPane.showMessageDialog(null, "Erro ao desconectar" + ex.toString());
43         }
44     }
45 }
46 }

```

Exemplo05 – Inclusão de Dados no Banco – prepareStatement

```

1  package teste;
2  import java.sql.Connection;
3  import java.sql.DriverManager;
4  import java.sql.PreparedStatement;
5  import java.sql.SQLException;
6  import javax.swing.JOptionPane;
7  public class AplicacaoIncluir2 {
8
9  private static Connection obterConexao() {...}
10
22
23 public static void main(String args[]){
24     Connection conexao = obterConexao();
25     PreparedStatement comando = null;
26     String nome = "Refrigerantes";
27     float promocao = 0;
28     float margem = 50;
29     try {
30         comando = conexao.prepareStatement("INSERT INTO GRUPOPRODUTO ( NOME, PROMOCAO, MARGEMPLUCRO ) VALUES (?, ?, ?)");
31         comando.setString(1, nome);
32         comando.setFloat(2, promocao);
33         comando.setFloat(3, margem);
34         comando.executeUpdate();
35         JOptionPane.showMessageDialog(null, "Inclusão realizada com sucesso");
36     } catch (SQLException ex) {

```

Exemplo06 – Alteração de Dados no Banco – preparedStatement

```

1  package teste;
2  import ...
7  public class AplicacaoAlterar1 {
8
9  private static Connection obterConexao() {...}
22
23 public static void main(String args[]){
24     Connection conexao = obterConexao();
25     PreparedStatement comando = null;
26
27     int codigo = 8; //verificar a existencia do codigo no banco
28     float promocao = 20;
29     float margem = 30;
30
31     try {
32         comando = conexao.prepareStatement("UPDATE grupoproduto SET promocao=?, margemlucro=? WHERE codigo=?");
33         comando.setFloat(1, promocao);
34         comando.setFloat(2, margem);
35         comando.setInt(3, codigo);
36         comando.executeUpdate();
37         JOptionPane.showMessageDialog(null, "Alteracao realizada com sucesso");
38     } catch (SQLException ex) {

```

Exemplo07 – Exclusão de Dados no Banco – uso do retorno do método executeUpdate()

```

1  package teste;
2  import ...
7  public class AplicacaoExcluir1 {
8
9  private static Connection obterConexao() {...}
22
23  public static void main(String args[]){
24      Connection conexao = obterConexao();
25      PreparedStatement comando = null;
26
27      int codigo = 7; //verificar a existencia do codigo no banco
28
29      try {
30          comando = conexao.prepareStatement("DELETE FROM grupoproduto WHERE codigo=?");
31          comando.setInt(1, codigo);
32          ↓
33          int cont = comando.executeUpdate();
34          JOptionPane.showMessageDialog(null, "Exclusao realizada com sucesso ["+cont+" excluido]");
35
36      } catch (SQLException ex) {

```

Exemplo08 – Seleção de Dados no Banco – recuperação de um registro específico

```

1  package teste;
2  import java.sql.Connection;
3  import java.sql.DriverManager;
4  import java.sql.PreparedStatement;
5  import java.sql.ResultSet;
6  import java.sql.SQLException;
7  import javax.swing.JOptionPane;
8  public class AplicacaoSelecao1 {
9      private static Connection obterConexao() {...}
22  public static void main(String args[]){
23      Connection conexao = obterConexao();
24      PreparedStatement comando = null;
25      int codigo = Integer.parseInt(JOptionPane.showInputDialog("Forneça o código a ser pesquisado"));
26      try {
27          comando = conexao.prepareStatement("SELECT * FROM grupoproduto WHERE codigo=?");
28          comando.setInt(1, codigo);
29          ResultSet resultado = comando.executeQuery();
30          if(resultado.next()){
31              JOptionPane.showMessageDialog(null, "Encontrado: "+resultado.getString("nome"));
32          }else{
33              JOptionPane.showMessageDialog(null, "Não encontrado");
34          }
35      } catch (SQLException ex) {

```

Exemplo09 – Seleção de Dados no Banco – recuperação de um conjunto de registros


```

1  package teste;
2  import ...
8  public class AplicacaoSelecao2 {
9  private static Connection obterConexao() {...}
22
23 public static void main(String args[]){
24     Connection conexao = obterConexao();
25     PreparedStatement comando = null;
26     try {
27         comando = conexao.prepareStatement("SELECT * FROM grupoproduto ORDER BY nome");
28         ResultSet resultado = comando.executeQuery();
29         while(resultado.next()){
30             System.out.println("Codigo: " +resultado.getInt("codigo"));
31             System.out.println("Nome: " +resultado.getString("nome"));
32             System.out.println("% Promocao: " +resultado.getFloat("promocao"));
33             System.out.println("% Margem lucro: " +resultado.getFloat("margemlucro"));
34             System.out.println("-----");
35         }
36         resultado.close();
37     } catch (SQLException ex) {

```

Exemplo10 – Executando comandos de DDL – método execute(sql)

```

1  package teste;
2  import ...
7  public class AplicacaoCreate {
8  private static Connection obterConexao() {...}
21
22  public static void main(String args[]){
23
24      Connection conexao = obterConexao();
25      Statement comando = null;
26      String sql = "CREATE TABLE teste (codigo SERIAL PRIMARY KEY, nome CHAR(40))";
27
28      try {
29          comando = conexao.createStatement();
30          comando.execute(sql);
31          JOptionPane.showMessageDialog(null, "Instrução executada com sucesso");
32      } catch (SQLException ex) {
33          JOptionPane.showMessageDialog(null, "Erro ao executar instrucao" + ex.toString());
34      } finally{

```