Desenvolvimento de Aplicações Web

Prof. Pedro Clarindo da Silva Neto

Parte VII

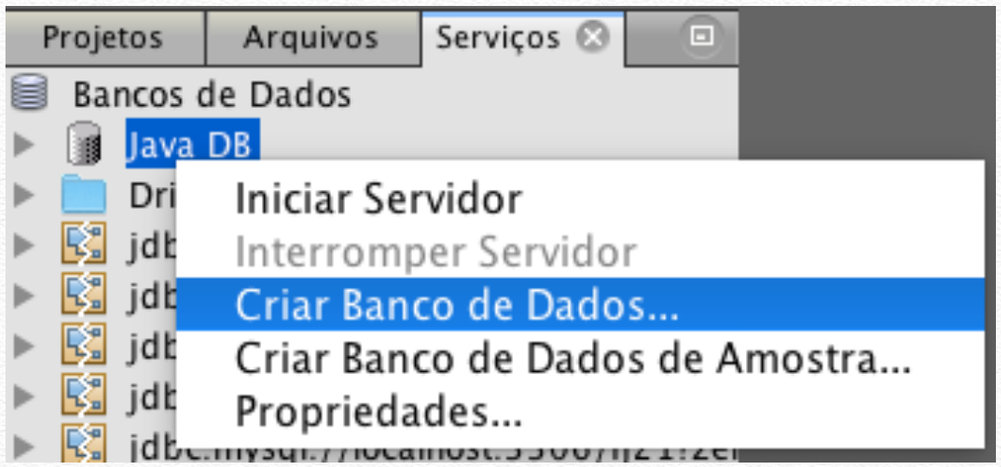
Servlet acessando banco de dados

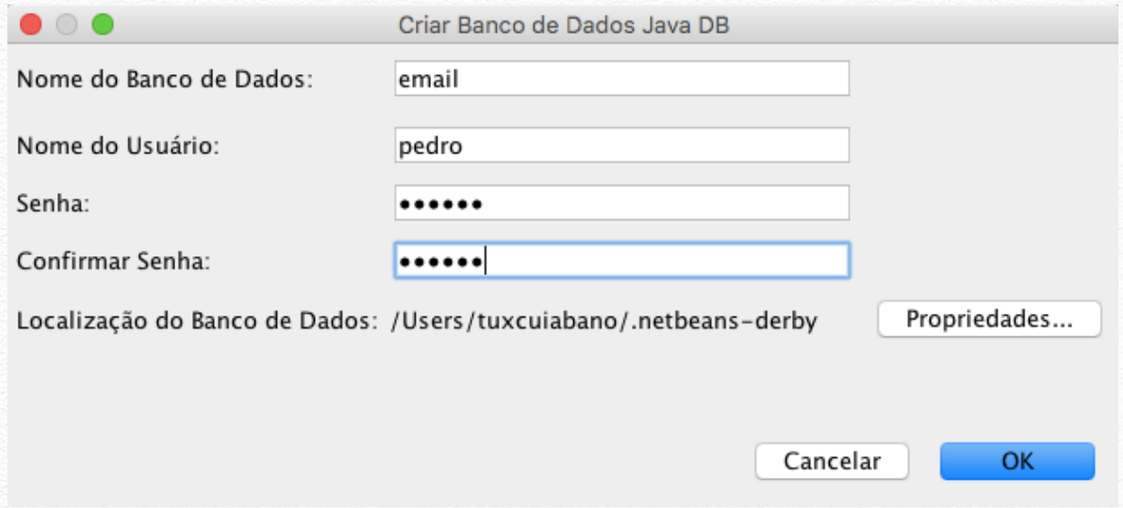
Antes de conectarmos um servlet a um banco de dados, é necessário que este banco seja criado. O NetBeans possui uma ferramenta para gerenciar bancos de dados que permite, por exemplo, criar um banco, criar suas tabelas, realizar consultas e inserir dados nas tabelas do banco. Além disso, as instalações padrões já vêm com um banco de dados chamado "Java DB" previamente instalado.

Neste material, iremos utilizar o banco "Java DB", mas a lógica dos programas é a mesma caso fosse utilizado outro banco, como MySQL por exemplo. É provável que o "Java DB" banco já esteja instalado e configurado em seu NetBeans.

Servlet acessando banco de dados

Para conferir, clique na aba “Serviços”, expanda a opção “Banco de Dados”, clique com o botão direito em “Java DB” e crie um novo banco de dados.

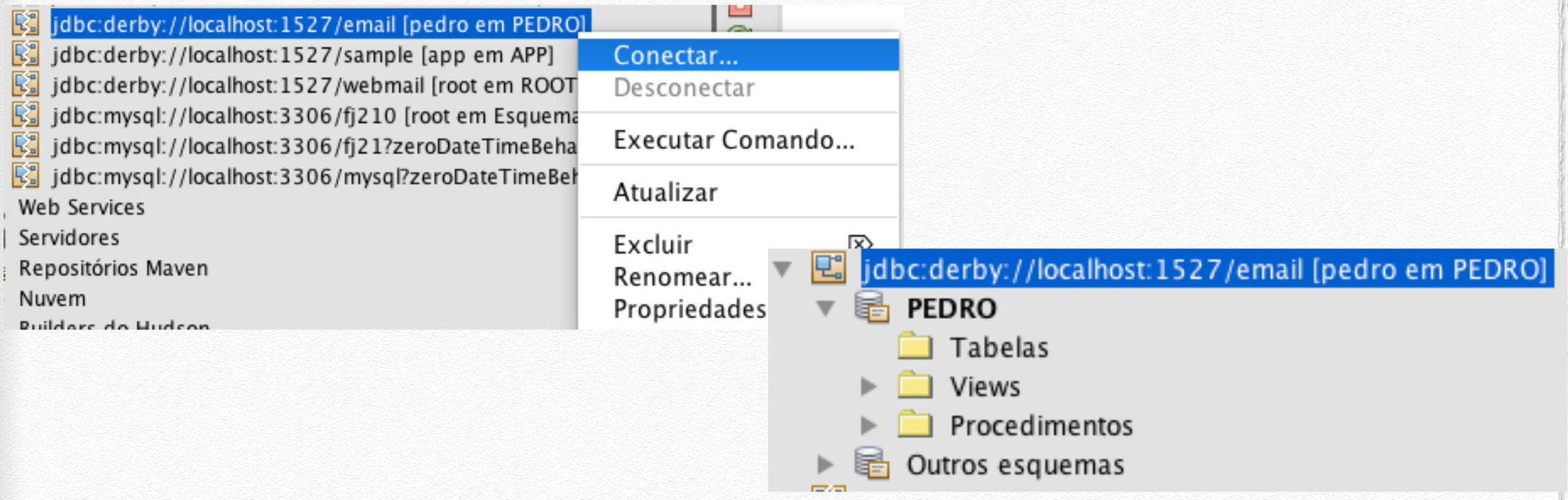


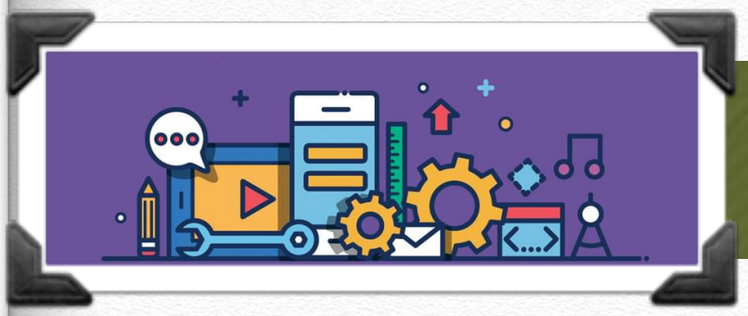


Em seguida, escolha um nome para o banco (por exemplo, “webmail”), um login e uma senha, que serão necessários para a conexão com o banco. O banco criado irá aparecer na aba “Serviços”.

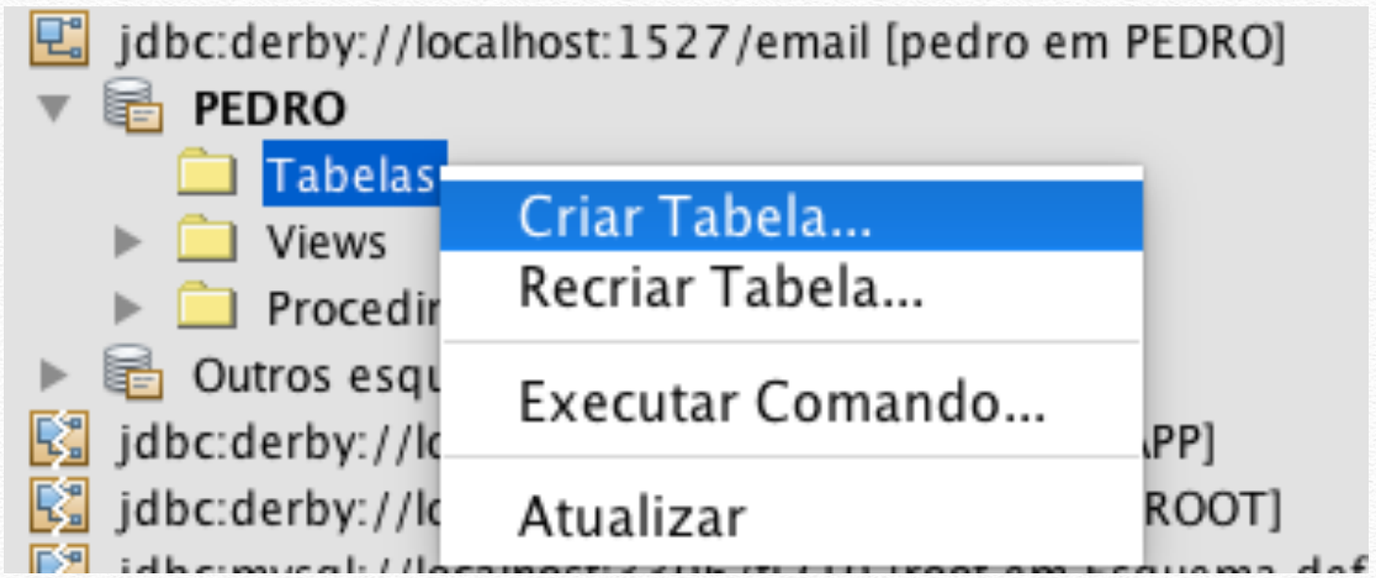
Servlet acessando banco de dados

Clique com o botão direito no banco, escolha a opção “Conectar ao banco” e depois expanda-o. Repare que agora é possível visualizar as tabelas do banco (por enquanto nenhuma tabela foi criada). Para criar uma tabela, clique com o botão direito em “Tabelas” e escolha a opção “Criar Tabela”.



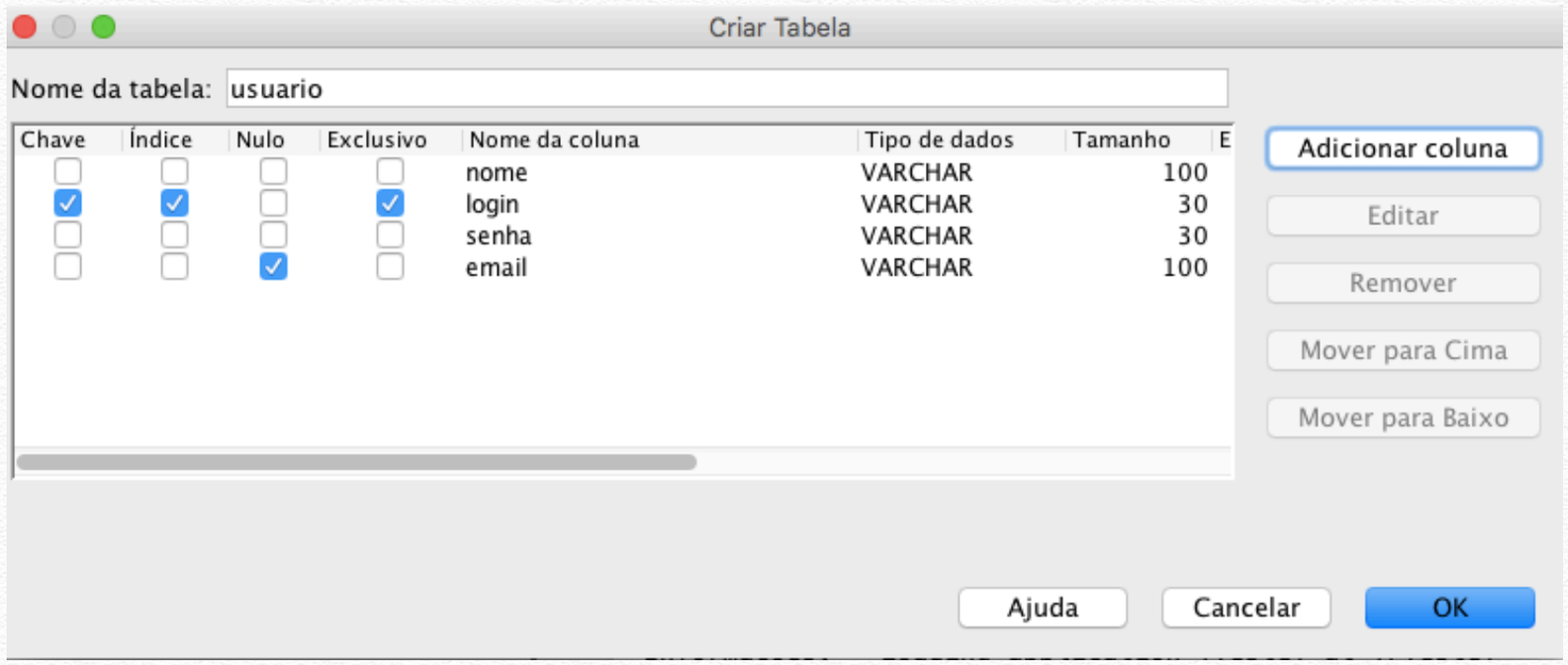


Servlet acessando banco de dados



O banco de dados para nosso webmail irá armazenar o nome, o login, a senha e um email alternativo de cada usuário (o email alternativo será necessário para que o usuário possa recuperar sua senha). A Figura a seguir mostra a criação destes campos na tabela “usuário”.

Servlet acessando banco de dados



Repare que o login é chave primária da tabela (não pode haver dois usuários com mesmo login, nem usuário sem login). Após a criação da tabela, se clicarmos com o botão direito sobre ela na aba "Serviços",podemos escolher a opção "Visualizar Dados". Esta opção nos mostra os dados já inseridos e permite inserirmos dados na tabela.

Conectando o servlet a um banco de dados

Para conectarmos a um banco de dados em Java, precisamos importar a biblioteca **java.sql.*** e inserir o trecho de código a seguir no local onde queremos fazer a conexão:

```
try (PrintWriter out = response.getWriter()) {
    /* TODO output your page here. You may use following sample code. */
    String driver = "org.apache.derby.jdbc.ClientDriver";
    String url = "jdbc:derby://localhost:1527/email";
    String username = "pedro";
    String password = "123456";


    try {
        Class.forName(driver);
        Connection connection = DriverManager.getConnection(url, username, password);
        Statement statment = connection.createStatement();
        connection.close();

    } catch (ClassNotFoundException cnfe) {

        out.println("Error loading driver" + cnfe);

    } catch (SQLException sqle) {
        out.println("Error with connection" + sqle);

    }
}
```

Cadastro de usuário no banco

Para que os usuários possam se inscrever em nosso webmail, devemos criar uma página de cadastro e apontar para ela na página inicial. A página deve conter um formulário onde o usuário preenche os dados necessários de acordo com o banco criado (em nosso caso, os dados são “nome”, “login”, “senha” e “email”).

É importante que o usuário possa inserir valores que caibam nos campos do banco de dados. Para isso, utilize o atributo “maxlength” da tag **<input />** para definir este número máximo de caracteres de acordo com o tamanho dos campos criados na tabela.

Cadastro de usuário no banco

O servlet de cadastro que irá receber os valores preenchidos pelo usuário deve se conectar ao banco de dados e enviar um comando SQL referente a uma inserção, como a seguir:

```
INSERT INTO usuario VALUES (‘Pedro Clarindo da Silva Neto’,
‘tuxcuiabano’, ‘123’, ‘pedro.neto@cba.ifmt.edu.br’)
```

Cadastro de usuário no banco

Para criarmos uma string em Java com este comando, precisamos recuperar os valores preenchidos pelo usuário no formulário. O código em Java deve ser semelhante ao trecho a seguir:

```
String nome = request.getParameter("nome");
String login = request.getParameter("login");
String senha = request.getParameter("senha");
String email = request.getParameter("email");
String insert = "INSERT INTO usuario VALUES('"+nome+"', '"+
        login+"', '"+senha+"', '"+email+"')";
statment.executeUpdate(insert);
```

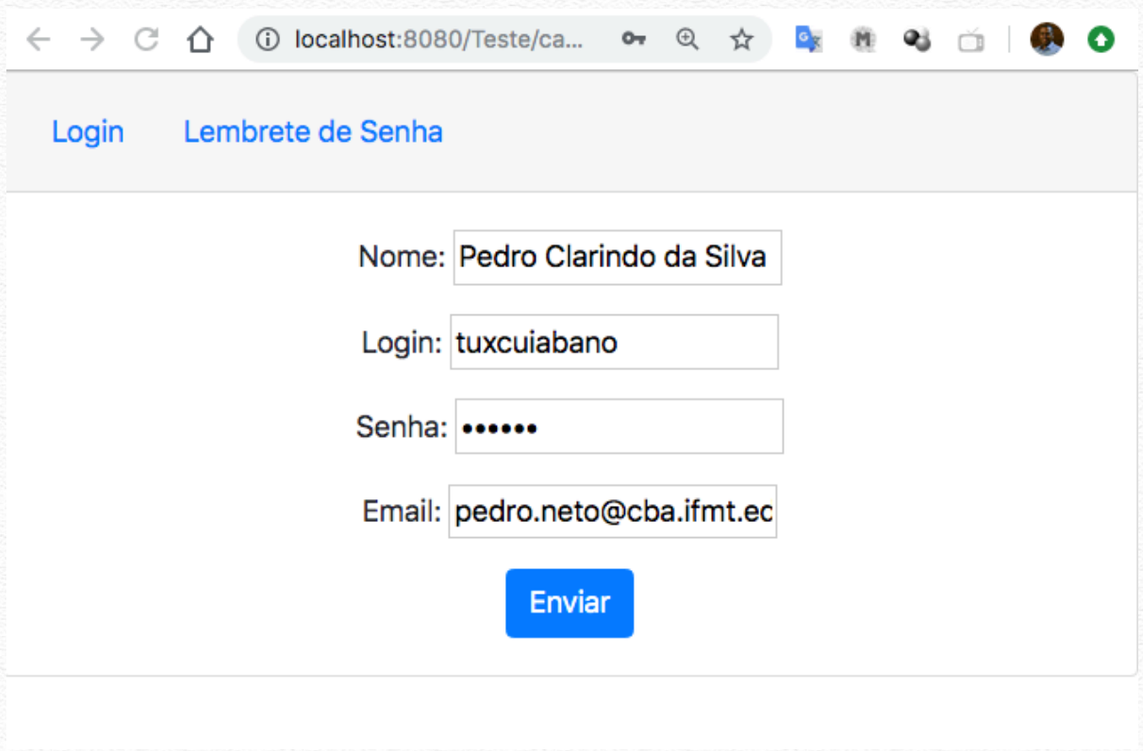
Cadastro de usuário no banco

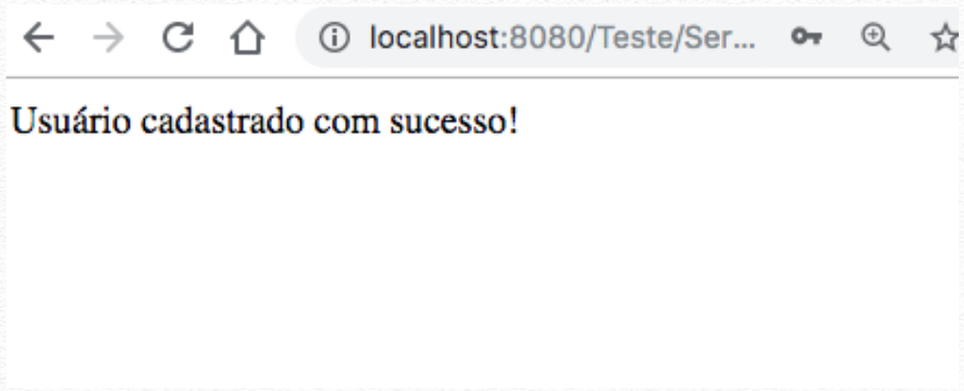
Para criarmos uma string em Java com este comando, precisamos recuperar os valores preenchidos pelo usuário no formulário. O código em Java deve ser semelhante ao trecho a seguir:

```
String nome = request.getParameter("nome");
String login = request.getParameter("login");
String senha = request.getParameter("senha");
String email = request.getParameter("email");
String insert = "INSERT INTO usuario VALUES('"+nome+"', '"+
        login+"', '"+senha+"', '"+email+"')";
statment.executeUpdate(insert);
```

Cadastro de usuário no banco

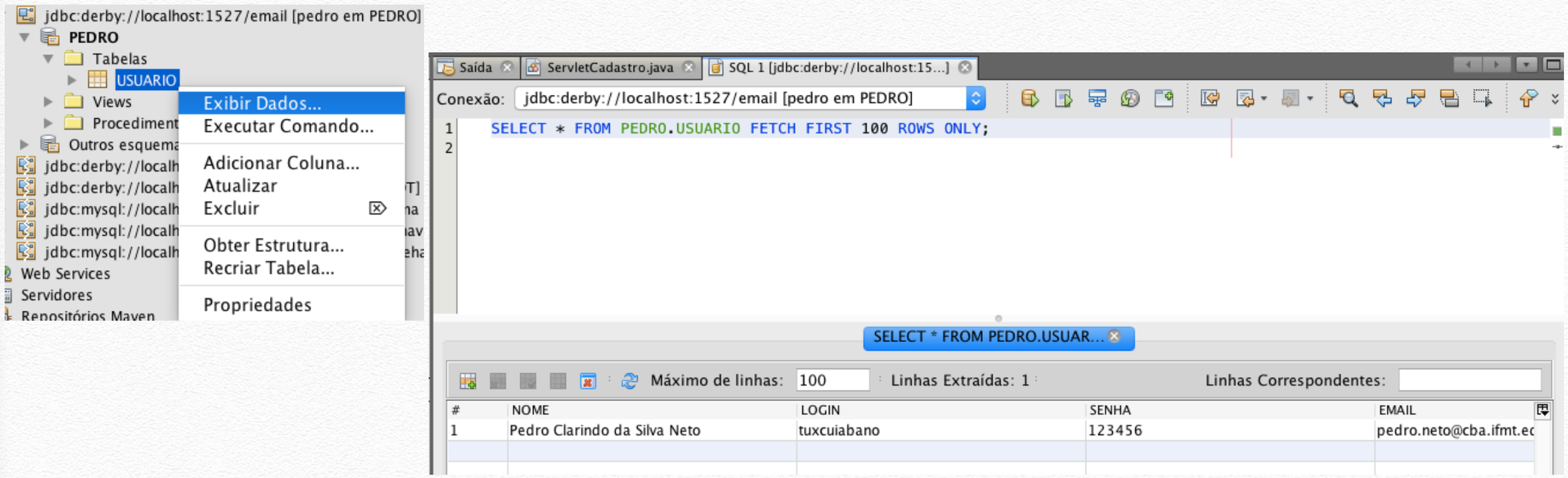
Repare que os parênteses e as aspas simples do comando SQL devem aparecer na string resultante. Após adaptar o código ao seu projeto, execute-o e teste se os usuários vão aparecendo no banco na medida em que forem sendo criados através do site.





Cadastro de usuário no banco

Para ver os dados é só ir na aba “Serviços", clicar com o botão direito na tabela “usuario" e escolher a opção “Exibir Dados”. Abrirá uma nova aba, mostrando os dados gravados na tabela.



Login

Após inserir alguns usuários no banco, podemos retomar nosso servlet de login criado no capítulo passado. O site de login envia ao servlet os valores de "login" e "senha" preenchidos pelo usuário. Devemos consultar o banco para verificar se o usuário existe e, neste caso, se sua senha está correta. O servlet deve se conectar ao banco de dados e enviar um comando SQL referente a uma consulta, como a seguir:

```
SELECT nome, senha FROM usuario WHERE login = 'tuxcuiabano'
```

Login

Para realizarmos esta consulta em Java, precisamos de um código semelhante ao trecho a seguir:

```
String login = request.getParameter("login");
String senha = request.getParameter("senha");
String query = "SELECT nome, senha FROM usuario WHERE login='"+login+"'";
ResultSet resultados = statement.executeQuery(query);
```

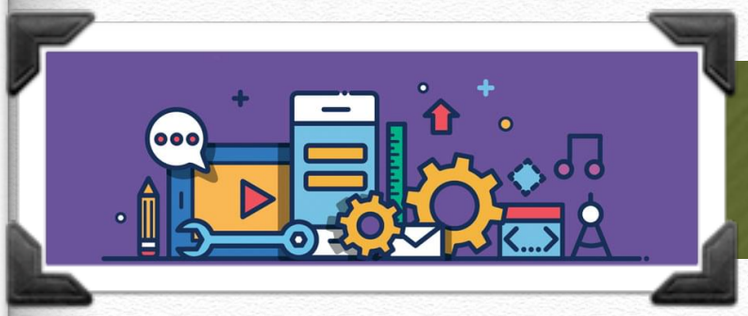
O objeto do tipo "ResultSet" contém o conjunto de linhas resultantes de uma consulta ao banco. Para iterarmos entre as linhas do conjunto de resultados, utilizamos o método "**resultados.next( )**".

Login

O trecho de código a seguir, por exemplo, imprimiria o nome de todos os usuários retornados pela consulta acima:

```
while(resultados.next()){
   out.println(resultados.getString("nome"));
}
```

Repare que “nome”, neste caso, se refere a uma coluna presente no conjunto de resultados retornado por nossa consulta. Entretanto, sabemos que o conjunto de resultados, em nosso caso, deverá conter zero ou um elemento (já que o campo “login” é chave primária de nossa tabela).



Login

Portanto, ao invés de iterar entre os usuários retornados, precisa- mos apenas verificar se algum usuário foi encontrado. O esqueleto do código ficaria assim:

```
String login = request.getParameter("login");
String senha = request.getParameter("senha");
String query = "SELECT nome, senha FROM usuario WHERE login='" +
        login + "'";
ResultSet resultados = statement.executeQuery(query);

if (resultados.next()) {
    String nome = resultados.getString("nome");
    String senha2 = resultados.getString("senha");

    if (senha2.equals(senha)) {
        //senha correta

    } else {
        //senha incorreta

    }

} else {

    //usuario não encontrado
}
```

Login

Agora que já sabemos conectar ao banco, adapte o servlet de login feito no capítulo passado para que ele verifique se o usuário e a senha estão realmente corretas, de acordo com os dados cadastros no banco.

Referências

Programação para internet. / Hilário Seibel Júnior. – Vitória: Ifes, 2010.

CAELUM, Java e Orientação a Objetos, Apostila. [Internet: http:// www.caelum.com.br/ downloads/apostila/caelum-java-objetos-fj11.pdf]. Acesso em 04/03/2009.

DEITEL, Harvey M.; DEITEL, Paul J., Java: Como Programar, São Paulo: Prentice-Hall, 2005.

GOODMAN, Danny, JavaScript a Bíblia, Ed. Campus, 2001

HALL, Marty; BROWN, Larry, Core Servlets e JavaServer Pages vo1 e vol 2, Ed. Ciência Moderna, 2005

HORSTMANN, Cay; CORNELL, Gary, Core Java 2, Fundamentos, São Paulo: Makron Books, Volume 1, 2000.

KURNIAWAN, Budi, Java para a Web com Servlets, JSP e EJB, Ed. Ciência Moderna, 2002

MUSCIANO, Chuck; KENNEDY, Bill, HTML: The definitive guide, Ed. Orelly, 1997

OLSON, Steven Douglas, Ajax com Java, Ed. Alta Books, 2007