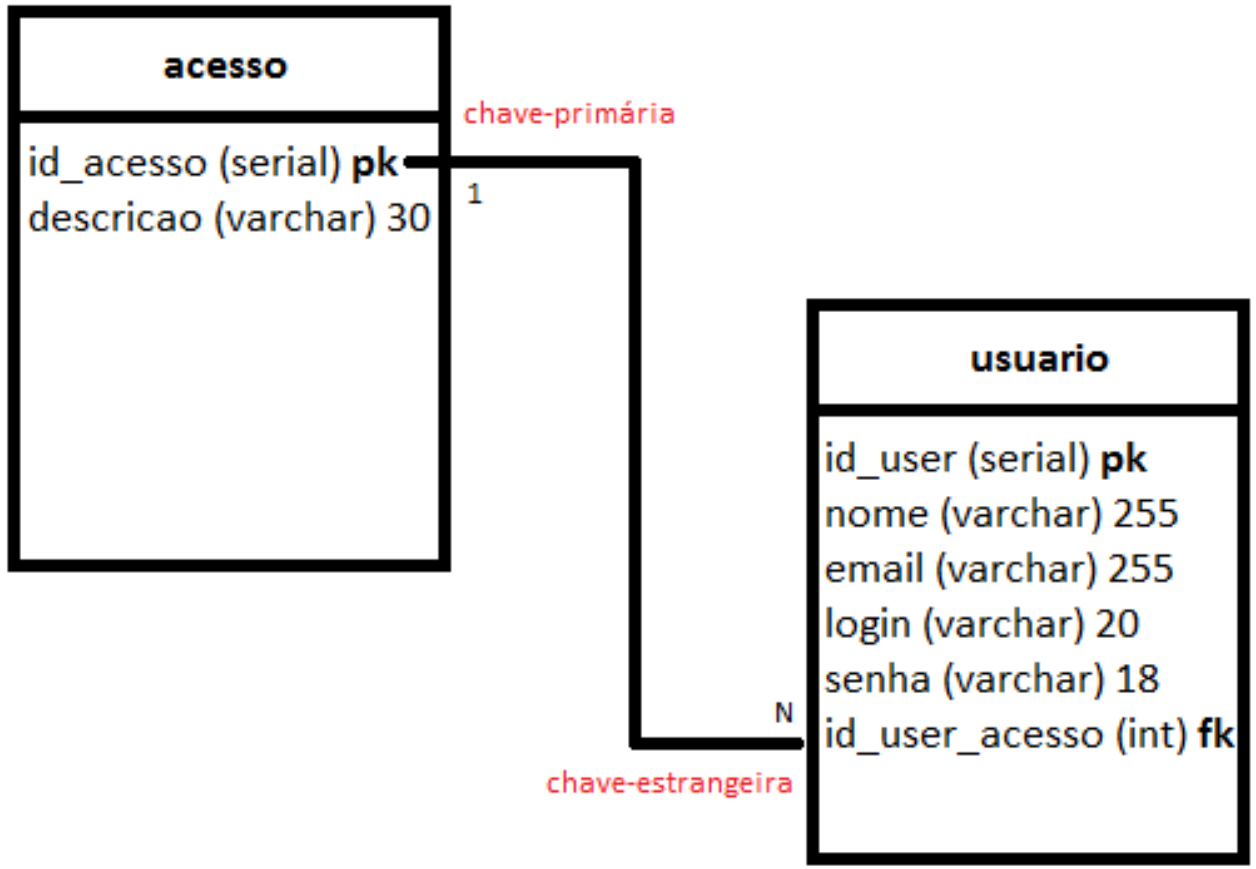
Curso de Java

aula 18

Prof. Anderson Henrique

JavaEE (Integrando com Banco de Dados)

Criar um banco de dados no postgreSQL com o nome: **projeto_java**



TECH
soluções em ti

Anteriormente, aprendemos a trabalhar com o JDBC, vamos criar uma classe Java responsável pela conexão, fechamento e obtenção da conexão estabelecida com a nossa Base de Dados.

Para isso, vamos criar um novo projeto Java Web e chamá-lo de **sistema_web**.

Dentro de Pacotes de Código-fonte, crie um pacote chamado **model**. Após isso, crie uma classe Java dentro deste pacote chamada **conexaoBancoDados**.

Esse é o código da classe Java **conexaoBancoDados**:

```java
package banco_dados;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class conexaoBancoDados {

    Connection conn;

    public boolean abrirConexao() throws ClassNotFoundException{
        String url = "jdbc:postgresql://localhost:5432/projeto_java";

        try{
            Class.forName("org.postgresql.Driver");
            conn = DriverManager.getConnection(url, "postgres", "root");
            return true;
        }catch(SQLException e){
            e.getMessage();
            return false;
        }
    }
```

A princípio vemos que esta classe se encontra no pacote **model**, logo após a declaração da classe temos a declaração de um atributo chamado **conn**.

Vemos o primeiro método chamado **abrirConexao( )** que tem por finalidade criar uma conexão com nosso Banco de Dados a atribuí-la no atributo **conn**. Retorna um valor booleano e lança exceções.

Veja o código a seguir, onde nos é apresentado o segundo método desta classe:

```
public boolean fecharConexao(){
    try{
        conn.close();
        return true;
    }catch(SQLException e){
        e.getMessage();
        return false;
    }
}
```

Este método é responsável por fechar a conexão que foi estabelecida com o nosso Banco de Dados, também retorna um valor booleano.

Nosso último método dessa classe é responsável por obter a conexão, retorna um objeto do tipo **Connection**:

```
    public Connection obterConexao(){
        return conn;
    }

}
```

Vamos criar uma classe que é responsável pela manipulação da entidade **acesso** no nosso Banco de Dados, também será criada no mesmo pacote **model**:

```java
package banco_dados;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Acesso {

    private Connection conn;
    private PreparedStatement statement;
    private ResultSet result;

    public void configurarConexao (Connection conn){
        this.conn = conn;
    }
```

Logo após a declaração da classe, temos a declaração de três objetos: **conn**, para conexão com o Banco de Dados; **statement**, utilizado na execução dos comandos SQL; e **result**, que conterá o resultado da execução de comandos de seleção de registros na tabela.

O primeiro método dessa classe, denominado **configurarConexao( )**, é responsável por atribuir ao objeto **conn** a referência ao Banco de Dados a que se teve uma conexão estabelecida.

Veja logo abaixo, a descrição do método **inserirRegistro**:

```
public boolean inserirRegistro(String descricao){
    String sql;
    try{
        sql = "insert into acesso(descricao)VALUES('"+descricao+"')";
        statement = conn.prepareStatement(sql);
        statement.executeUpdate();
        return true;
    }catch(SQLException e){
        System.out.println(e.getMessage());
        return false;
    }
}
```

Esse método recebe como **parâmetro** uma cadeia de caracteres que representa a descrição do acesso. Então, uma instrução **SQL** que utiliza o comando **INSERT** é definida e depois executada.

Para a execução dessa instrução, utilizamos o objeto **statement**, que é declarado como sendo do tipo **PreparedStatement**. Por meio do objeto **conn**, atribuímos a **statement** o valor retornado pelo método. Após isso, executamos a instrução através do método **executeUpdate( )**.

```java
public boolean alterarRegistro(int id, String descricao){
    String sql;
    try{
        sql = "update acesso set descricao = '"+descricao+"' where "
                + "id_acesso = "+id;
        statement = conn.prepareStatement(sql);
        statement.executeUpdate();
        return true;
    }catch(SQLException e){
        System.out.println(e.getMessage());
        return false;
    }
}
```

Esse método tem uma estrutura bastante similar a anterior. A diferença entre eles reside no fato de **alterarRegistro( )** fazer uso de dois parâmetros e de a instrução SQL também ser outra.

Os parâmetros se referem a chave-primária do registro que será alterado e ao novo valor dessa descrição.

```java
public boolean excluirRegistro(int id){
    String sql;
    try{
        sql = "delete from acesso where id_acesso = "+id;
        statement = conn.prepareStatement(sql);
        statement.executeUpdate();
        return true;
    }catch(SQLException e){
        System.out.println(e.getMessage());
        return false;
    }
}
```

Esse método é utilizado na exclusão de algum acesso. Ele recebe como parâmetro apenas a chave primária do registro que será excluído. A partir de então o comando SQL se responsabiliza por criar a expressão e o método **executeUpdate( )**, por executar esse comando.

TECH
soluções em ti

```java
public ResultSet listarRegistros(){
    String sql;
    try{
        sql = "select * from acesso order by id_acesso desc";
        statement = conn.prepareStatement(sql);
        result = statement.executeQuery();
        return result;
    }catch(SQLException e){
        System.out.println(e.getMessage());
        return null;
    }
}
```

Por fim, esse método tem a função de retornar um conjunto de registros da tabela acesso. Esse método retorna um objeto do tipo **ResultSet**.

Com as classes preparadas, vamos aprender como utilizá-las.

Para testarmos as classes: **conexaoBancoDados** e **Acesso** e entendermos como elas funcionam na nossa aplicação, vamos criar página HTML chamada **view_principal**, **view_inserir**, **view_erro**.

Também, vamos criar um novo pacote chamado **controller**, dentro dele vamos criar um Servlet chamado: **GerenciaAcao**.

E por fim, vamos criar os arquivos JSP para a ação: **insereDados.jsp** e **listaDados.jsp**.

Abaixo, segue o código da página HTML chamada **view_principal.html**:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Gerenciando Acesso</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <h2>Gerenciamento da Tabela Acesso</h2>
        <fieldset>
            <legend>Escolha uma das opções abaixo</legend>
            <form method="get" action="GerenciaAcao">
                <input type="radio" name="Acao" value="1">Inserir
                <input type="radio" name="Acao" value="2">Listar
                <input type="submit" value="Escolher">
            </form>
        </fieldset>
    </body>
</html>
```

TECH
soluções em ti

Abaixo, segue o código da página HTML chamada **view_inserir.html**:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Formulário Inserir Dados</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <h2>Inserir Acesso</h2>
        <fieldset>
            <legend>Preencha o campo abaixo</legend>
            <form method="get" action="insereDados.jsp">
                Descição do Acesso
                <input type="text" name="strAcesso" required>
                <input type="submit" value="Inserir">
            </form>
        </fieldset>
        <p>
            <a href="view_principal.html">Menu Principal</a>
        </p>
    </body>
</html>
```

Abaixo, segue o código da página HTML chamada **view_erro.html**:

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Erro</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <h2>**Erro**</h2>
        <h3>Selecione uma opção válida!</h3>
        <a href="view_principal.html">Menu Principal</a>
    </body>
</html>
```

TECH
soluções em ti

Abaixo, segue o código do Servlet chamado: **GerenciaAcao**:

```java
package controller;

import java.io.IOException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class GerenciaAcao extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        int intAcao;
        String urlInserir = "/view_inserir.html";
        String urlListar = "/listaDados.jsp";
        String urlErro = "/view_erro.html";

        if (request.getParameter("Acao") != null) {
            intAcao
                    = Integer.parseInt(request.getParameter("Acao"));
        } else {
            intAcao = 0;
        }
        RequestDispatcher despachante;
        switch (intAcao) {
            case 1:
                despachante
                        = getServletContext().getRequestDispatcher(urlInserir);
                break;
            case 2:
                despachante
                        = getServletContext().getRequestDispatcher(urlListar);
                break;
            default:
                despachante
                        = getServletContext().getRequestDispatcher(urlErro);
        }
        despachante.forward(request, response);

    }
}
```

TECH
soluções em ti

Abaixo, segue o código da página JSP chamada: **insereDados.jsp**:

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Insere Dados</title>
    </head>
    <body>
        <h2>Insere Dados</h2>
        <%
            String strAcesso;
            if(conexao.abrirConexao()){
                strAcesso = request.getParameter("strAcesso");
                acesso.configurarConexao(conexao.obterConexao());
                if(acesso.inserirRegistro(strAcesso)){
                    out.println("<p>Tipo de Acesso cadastrado com sucesso!</p>");
                }else{
                    out.println("<p>Erro ao tentar cadastrar Acesso</p>");
                    conexao.fecharConexao();
                }
            }else{
                out.println("<p>Falaha na conexão com o Banco de Dados!</p>");
            }
        %>
        <p><a href="view_principal.html">Menu Principal</a></p>
    </body>
</html>
```

JavaEE (Componente JavaBean)

No início do código acima, temos duas linhas cuja sintaxe não tinha sido apresentada. Trata-se do elemento sintático do JSP, onde existe uma ação denominada **useBean**. Ela permite que uma classe Java pura, como as duas que criamos, possa ser utilizada em aplicações web.

Esse tipo de classe é comumente denominado componente **JavaBean**, ou simplesmente **bean**.

Para o uso da ação **useBean**, é necessário configurar três parâmetros:

1) **id:** trabalha como uma instanciação da classe Java, como na declaração de um objeto;
2) **class:** nome do pacote e da classe;
3) **scope:** define a abrangência da instância da classe. Pode conter os seguintes valores:
   - I. **page:** escopo de toda a página JSP em que o bean foi declarado;
   - II. **request:** o bean instanciado pode ser utilizado por qualquer página JSP capaz de atender à mesma solicitação do cliente;
   - III. **session:** qualquer página JSP que compartilha a mesma seção da página JSP responsável pela criação do bean tem acesso a este valor;
   - IV. **application:** o bean criado pela página JSP pode ser utilizado em todas as páginas da aplicação à qual ela pertence.

O núcleo da página é o código JSP que abre uma conexão com o Banco de Dados e insere um registro na tabela **acesso**.

Se for possível abrir a conexão, ela é atribuída ao objeto **acesso**. Por meio de uma chamada ao método **inserirRegistro( )** desse objeto, adicionamos um acesso que será digitado no formulário do cliente à tabela **acesso**.

Abaixo, segue o código da página JSP, chamada **listaDados.jsp**:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Lista Acesso</title>
    </head>
    <body>
        <h2>Listar Dados</h2>
        <table style="width: 40%; text-align: center">
            <tr style="background: #333; color: #FFF">
                <td>ID</td>
                <td>Descrição</td>
            </tr>
```

Também utilizamos o componente JavaBeans, permitindo que as classes Java sejam utilizadas na nossa aplicação.

```
        <%@page import="java.sql.*" %>
        <%
            ResultSet rs;
            if(conexao.abrirConexao()){
                acesso.configurarConexao(conexao.obterConexao());
                rs = acesso.listarRegistros();
                while(rs.next()){
        %>
        <tr>
            <td><% out.println(rs.getString(1));%></td>
            <td><% out.println(rs.getString(2));%></td>
        </tr>
        <%
                }
            }else{
                out.println("<p>Falha na conexão com o Banco de Dados!</p>");
            }
        %>
    </body>
</html>
```

Utilizamos a diretiva **page** para importar a classe java que precisamos para criar um objeto do tipo **ResultSet** denominado **rs**.

Ainda precisamos das rotinas de excluir e alterar, para isso vamos fazer uma pequena alteração no arquivo **listaDados.jsp**:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Lista Acesso</title>
    </head>
    <body>
        <h2>Listar Dados</h2>
        <table style="width: 40%; text-align: center">
            <tr style="background: #333; color: #FFF">
                <td>ID</td>
                <td>Descrição</td>
                <td>Ações</td>
            </tr>
        <%@page import="java.sql.*" %>
        <%
            ResultSet rs;
            if(conexao.abrirConexao()){
                acesso.configurarConexao(conexao.obterConexao());
                rs = acesso.listarRegistros();
                while(rs.next()){
        %>
```

```
        <tr>
            <td><% out.println(rs.getString(1));%></td>
            <td><% out.println(rs.getString(2));%></td>
            <td>
                <% out.println("<a href='?id="+rs.getString(1)+"'>Editar</a>");%>
                <% out.println("<a href='excluiDados.jsp?id="+rs.getString(1)+"'>Excluir</a>");%>
            </td>
        </tr>
        <%
                }
            }else{
                out.println("<p>Falha na conexão com o Banco de Dados!</p>");
            }
        %>
        <p><a href="view_principal.html">Menu Principal</a></p>
    </body>
</html>
```

Adicionamos mais uma coluna na primeira linha da tabela com o conteúdo **Ação** e também adicionamos uma coluna na segunda linha contendo dois links, sendo passado como parâmetro a chave primária do registro.

No final acrescentamos um link para retornar para a página principal.

Segue abaixo o código da página JSP **excluiDados** chamada no link **Excluir** na tabela **Lista Acesso**:

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Excluir Dados</title>
    </head>
    <body>
        <%
            int intId;
            if(request.getParameter("id") != null){
                intId =
                   Integer.parseInt(request.getParameter("id"));
            }else{
                intId = 0;
            }

            if(conexao.abrirConexao()){
                acesso.configurarConexao(conexao.obterConexao());
                if(acesso.excluirRegistro(intId)){
                    out.println("<p>Tipo de Acesso excluído com sucesso!</p>");
                }else{
                    out.println("<p>Erro ao tentar excluir Acesso</p>");
                    conexao.fecharConexao();
                }
            }else{
                out.println("<p>Falha na conexão com o Banco de Dados!</p>");
            }
        %>
        <p><a href="view_principal.html">Menu Principal</a></p>
    </body>
</html>
```

TECH soluções em ti

No código acima também utilizamos o **JavaBeans** para permitir que as classes Java pura possam ser usadas na nossa página **JSP**.

Por fim, iremos criar a rotina para alterar a descrição do acesso pela chave primária do registro, devemos acrescentar um método na classe **Acesso**, chamado **listarPeloId( )**.

```
public ResultSet listarPeloId(int intId){
    String sql;
    try{
        sql = "select * from acesso where id_acesso = "+intId;
        statement = conn.prepareStatement(sql);
        result = statement.executeQuery();
        return result;
    }catch(SQLException e){
        System.out.println(e.getMessage());
        return null;
    }
}
```

TECH
soluções em ti

Na página **listaDados.jsp**, no link **Editar**, vamos inserir uma página no atributo **href** chamada **formEditar.jsp**, ela já passa como parâmetro a **chave primária** do registro que deverá ser alterado.

```
<td>
    <% out.println("<a href='formEditar.jsp?id="+rs.getString(1)+"'>Editar</a>");%>
    <% out.println("<a href='excluiDados.jsp?id="+rs.getString(1)+"'>Excluir</a>");%>
</td>
```

A página **formEditar.jsp** também fará uso do recurso **JavaBeans** por ser necessário utilizar as **classes Java** para **conexão** com o Banco de Dados e a classe **Acesso** que implementa o método **listarPeloId**( ).

Segue abaixo o código da página formEditar.jsp:

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Formulário Edição</title>
    </head>
    <body>
        <%
           int intId;
           if(request.getParameter("id") != null){
               intId =
                 Integer.parseInt(request.getParameter("id"));
           }else{
               intId = 0;
           }
        %>
        <%@page import="java.sql.*"%>
        <%
            ResultSet rs;
            if(conexao.abrirConexao()){
                acesso.configurarConexao(conexao.obterConexao());
                rs = acesso.listarPeloId(intId);
                if(rs.next()){
        %>
```

Perceba que no código acima, as estruturas de decisão ainda se encontram abertas, isso se deve porque o nosso formulário que permitirá a edição deve se encontrar dentro dessas estruturas, como nos mostra o código a seguir:

```
        <h2>Formulário Edição de Acesso</h2>
        <fieldset>
            <legend>Preencha o campo abaixo</legend>
            <form method="get" action="editaDados.jsp">
                Descrição do Acesso
                <input type="text" name="idAcesso" value="<%= rs.getString(1)%>">
                <input type="text" name="strAcesso" required value="<%= rs.getString(2)%>">
                <input type="submit" value="Alterar">
            </form>
        </fieldset>
        <%
                }
            }
        %>
    </body>
</html>
```

O atributo action desse formulário invoca a página **editaDados.jsp**, é será nessa página que o registro, de fato, será alterado.

Para finalizar, esse é o código da página JSP chamada **editaDados**:

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="conexao" scope="page" class="model.conexaoBancoDados"/>
<jsp:useBean id="acesso" scope="page" class="model.Acesso"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Edita Dados</title>
    </head>
    <body>
        <h2>Editar Dados</h2>
        <p><a href="view_principal.html">Menu Principal</a></p>
        <%
            int id;
            String descricao;
            id = Integer.parseInt(request.getParameter("idAcesso"));
            descricao = request.getParameter("strAcesso");
            if (conexao.abrirConexao()) {
                acesso.configurarConexao(conexao.obterConexao());
                if (acesso.alterarRegistro(id, descricao)) {
                    out.println("<p>Tipo de Acesso editado com sucesso!</p>");
                }else{
                    out.println("<p>Erro ao tentar editar Acesso</p>");
                    conexao.fecharConexao();
                }
            }
        %>
    </body>
</html>
```

Prosseguiremos no próximo slide... Com JEE (JSP, Servlets, JavaBeans, Integração com o Banco de Dados, Login de Acesso)

Professor: Anderson Henrique

Programador nas Linguagens Java e PHP