Maestro-Detalle

Facultad de Ingeniería y Ciencias Naturales
Ingeniería en Sistemas Computacionales
Desarrollo del Software I Guía 8

I- Objetivos

- Conocer los formularios maestro-detalle
- Aplicar conocimientos adquiridos sobre maestros [CRUD]
- Crear aplicaciones maestro-detalle

II- Contenido

Introducción

Un maestro detalle es el que posee un área maestra, y un área de detalle es desplegada basada en el registro seleccionado en el área maestra. Un área maestra puede ser un formulario, cuadricula, lista o árbol de elementos, y el área de detalle puede ser también un formulario, cuadricula, lista o árbol de elementos ubicado típicamente debajo del área maestra o a continuación del área maestra. Seleccionando un elemento de la lista maestra hace que se muestre el detalle de ese elemento publicado en el área de detalle.

Un ejemplo de relación maestro detalle es: un conjunto de órdenes y un conjunto de items pertenecientes a cada una de las órdenes. Una aplicación puede usar la relación maestro detalle para permitir a los usuarios navegar a través de los datos de la orden y ver el detalle de los items que están relacionados únicamente con la orden seleccionada.

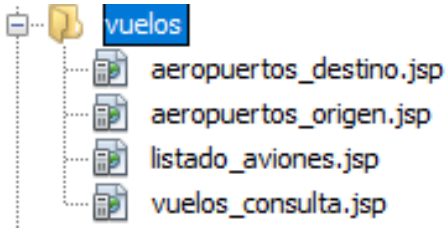
Requerimientos:

- ✓ Clases de lógica de negocios
- ✓ Sesiones y permisos
- ✓ Formularios de mantenimiento [CRUD]

Ejemplo 1. Aplicación Maestro-Detalle de Registro de Vuelos

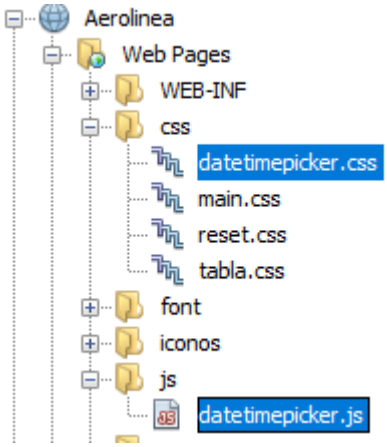
Paso 1.

Crear los siguientes archivos a continuación señalados en el proyecto AerolineaProject creado en la guía 6. **Preste especial atención a los comentarios en el código para entenderlo mejor**



Luego descargar los archivos **datetimepicker.css** y **datetimepicker.js** desde el aula virtual, para poder agregar el control de fecha y hora, para los controles que lo requieran. El archivo **datetimepicker.css** agregarlo a la carpeta css del proyecto, y el archivo **datetimepicker.js** agrégalo a la carpeta js del proyecto, si no existe la carpeta créela, y luego agregue el archivo en ella.

La estructura debe quedar así:



Y por último agregar al archivo **_top.jsp** las siguientes líneas al final de la etiqueta head:

```
<link rel="stylesheet" type="text/css" href="css/datetimepicker.css"/>
<script type="text/javascript" src="js/datetimepicker.js"></script>
```

Paso 2. Actualizar la clase Tabla.java

La clase Tabla.java debe ser actualizada con la versión más nueva

Clase: Tabla.java; Paquete: procesos

Descargar del Aula virtual o Ir a https://www.dropbox.com/sh/tyrc8o2od391isg/DH44jPiC7g

Y descargar el archivo Tabla.java, luego reemplazar la clase Tabla.java en el paquete procesos

Paso 3. Crear interfaz Maestro-Detalle

A continuación se crea la página principal para el registro de vuelos disponibles, en ella se podrá buscar el origen, destino y avión que componen un vuelo y también asignar una fecha y hora de salida.

Página: vuelos_consulta.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@include file="../_top.jsp"%>
<script>
function abrirVentana (URL){
   //funcion javascript para abrir un subventana para realizar
   //busquedas, se le pasa la pagina a mostrar como parametro

window.open(URL,"ventana1","width=700,height=400,scrollbars=YES,statusbar=YES,top=150,left=300")
}
</script>
```

```
    <h1>Registro de Vuelos</h1><br><br>
    <form name="main" method="post"
action="${pageContext.servletContext.contextPath}/Vuelos">
        <!--<input type="hidden" name="sw_nuevo" value="1"/>-->
        <br>
        Seleccione Aeropuerto Origen
        <input type="text" id="txtIdorigen" name="txtIdorigen"  value="${idorigen}" size="3"
readonly="readonly">
        <input type="text" id="txtOrigen" name="txtOrigen" size="50" value="${origen}"
readonly="readonly">
        <input type="button" class="boton" value="..."
onclick="abrirVentana('${pageContext.servletContext.contextPath}/Vuelos?accion=listado_origen');">
        <br><br><br><br>
        <hr>
        <br>
        Fecha y Hora:
        <input class="datepicker" type="text" name="txtFecha" size="25">
        Avion:
        <input type="text" name="txtIdavion" id="txtIdavion" size="2" readonly="readonly">
        <input type="text" name="txtAvion" id="txtAvion" readonly="readonly">
        <input type="button" value="..." class="boton"
onclick="abrirVentana('${pageContext.servletContext.contextPath}/Vuelos?accion=listado_avion');">
        Destino :
        <input type="text" name="txtIddestino" id="txtIddestino" size="2" readonly="readonly">
        <input type="text" name="txtDestino" id="txtDestino" readonly="readonly">
        <input type="button" value="..." class="boton"
onclick="abrirVentana('${pageContext.servletContext.contextPath}/Vuelos?accion=listado_destino');"
>
        <input type="submit" value="Agregar" class="boton">
    </form><br><br>
    <c:if test="${resultado!=null}">
      <c:if test="${resultado==1}">
        <p style="color:darkgreen; font-size: 15px; text-align: center"><strong>Operación realizada
correctamente.</strong></p>
      </c:if>
      <c:if test="${resultado==0}">
        <p style="color:darkred; font-size: 15px; text-align: center"><strong>La operación no se
realizó.</strong></p>
      </c:if>
    </c:if>
    ${tabla}
    <script>
      window.onload = function() {
        //inicializamos el control de fecha
        var dtp = new DateTimePicker('.datepicker', {
            timePicker: true, // activamos la selección de hora
            format: 'd/m/Y H:i' //formato de fecha y hora
```

```
                });
        };
        //funcion que se llamará al seleccionar el origen desde la ventana
        function setDataOrigen(idorigen, origen) {
            document.getElementById("txtIdorigen").value = idorigen;
            document.getElementById("txtOrigen").value = origen;
        }
        //funcion que se llamará al seleccionar el destino desde la ventana
        function setDataDestino(iddestino, destino) {
            document.getElementById("txtIddestino").value = iddestino;
            document.getElementById("txtDestino").value = destino;
        }
        //funcion que se llamará al seleccionar el avion desde la ventana
        function setDataAvion(idavion, avion) {
            document.getElementById("txtIdavion").value = idavion;
            document.getElementById("txtAvion").value = avion;
        }
    </script>
<%@include file="../_down.jsp"%>
```

Paso 4. Crear página de búsqueda para orígenes

A continuación se creará una página, que será llamada de manera emergente sobre la ventana principal de registro para realizar la búsqueda del aeropuerto-origen a ser procesado

Aeropuertos Origen

Idaeropuerto	Aeropuerto	Pais	Ciudad
50	BOSTON AIRPORT	ESTADOS UNIDOS	MASS
51	ATLANTA AIRPORT	ESTADOS UNIDOS	ATL
49	AEROPUERTO EL SALVADOR	EL SALVADOR	SAN SALVADOR
52	AEROPUERTO DE COSTA RICA	COSTA RICA	SAN JOSE

Resultado Aeropuertos

Página: aeropuertos_origen.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" type="text/css" href="css/reset.css" />
    <link rel="stylesheet" type="text/css" href="css/main.css" media="screen" />
    <link rel="stylesheet" type="text/css" href="css/tabla.css" media="screen" />
    <style>
      #table01 td{ padding-top: 8px; cursor: pointer}
    </style>
    <title>Origenes</title>
  </head>
  <body>
    <div id="contenido" style="padding: 10px">
    <h1>Aeropuertos Origen</h1>
    ${tabla}
  <script>
    //funcion javascript que se ejecuta al hacer click en una fila
    //recibe un elemento de tipo fila como parametro: row
    function _Seleccionar_(row){

      //recupera el idorigen de la fila, en la celda 0
      var idorigen = row.cells[0].innerHTML;
      //recupera nombre del origen de la fila, en la celda 1
      var origen = row.cells[1].childNodes[0].innerHTML;

      //redirecciona hacia vuelos.jsp con los valores obtenidos
      //de idorigen y origen
      window.opener.location.href=
      "${pageContext.servletContext.contextPath}/Vuelos?idorigen="+idorigen+"&origen="+origen;
      //cierra la ventana
      window.close();
      return false;
    }
  </script>
  </div>
  </body>
</html>
```

Paso 5. Crear página búsqueda de destinos

Aeropuertos Destino

Idaeropuerto	Aeropuerto	País	Ciudad
50	BOSTON AIRPORT	ESTADOS UNIDOS	MASS
51	ATLANTA AIRPORT	ESTADOS UNIDOS	ATL
49	AEROPUERTO EL SALVADOR	EL SALVADOR	SAN SALVADOR
52	AEROPUERTO DE COSTA RICA	COSTA RICA	SAN JOSE

Resultado Aeropuertos

Página: aeropuertos_destino.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" type="text/css" href="css/reset.css" />
    <link rel="stylesheet" type="text/css" href="css/main.css" media="screen" />
    <link rel="stylesheet" type="text/css" href="css/tabla.css" media="screen" />
    <style>
      #table01 td{ padding-top: 8px; cursor: pointer}
    </style>
    <title>Destinos</title>
  </head>
  <body>
    <div id="contenido" style="padding: 10px">
    <h1>Aeropuertos Destino</h1>
    ${tabla}
  <script>
    //funcion javascript que se ejecuta al hacer click en una fila
    //recibe un elemento de tipo fila como parametro: row
    function _Seleccionar_(row){

      //recupera el idorigen de la fila, en la celda 0
      var iddestino = row.cells[0].innerHTML;
```

```
        //recupera nombre del origen de la fila, en la celda 1
        var destino = row.cells[1].childNodes[0].innerHTML;

        // se manda a llamar la función desde la ventana padre que llamó esta ventana
        window.opener.setDataDestino(iddestino, destino);

        //cierra la ventana
        window.close();
    }
  </script>
  </div>
  </body>
</html>
```

Paso 6. Crear página para búsqueda de aviones

Listado de Aviones

Idavion	Avion	Capacidad
2	BOEING 777	200
3	KC-135	150
4	707-320	175
5	AIRBUS MRTT	100
Resultado Aeropuertos		

Página: listado_aviones.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" type="text/css" href="css/reset.css" />
    <link rel="stylesheet" type="text/css" href="css/main.css" media="screen" />
    <link rel="stylesheet" type="text/css" href="css/tabla.css" media="screen" />
    <style>
      #table01 td{ padding-top: 8px; cursor: pointer}
    </style>
```

```
    <title>Aviones</title>
  </head>
  <body>
    <div id="contenido" style="padding: 10px">
    <h1>Listado de Aviones</h1>
    ${tabla}
  <script>
    //funcion javascript que se ejecuta al hacer click en una fila
    //recibe un elemento de tipo fila como parametro: row
    function _Seleccionar_(row){
      //recupera el idavion de la fila, en la celda 0
      var idavion = row.cells[0].innerHTML;
      //recupera descripcion del avion de la fila, en la celda 1
      var avion = row.cells[1].childNodes[0].innerHTML;
      //asigna a las cajas de texto de la ventana padre los valores
      //obtenidos
      window.opener.setDataAvion(idavion, avion);
      //cierra la ventana
      window.close();
    }
  </script>
  </div>
  </body>
</html>
```

Paso 7. Crear el Servlet Vuelos.

Se créa este servlet para manejar todas las operaciones de vuelos, incluídos los listados de aviones, aeropuertos de origen, y destino. Debe quedar así:

Servlet: Vuelos.java

```
package com.aerolinea.control;

import com.aerolinea.conexion.Conexion;
import com.aerolinea.conexion.ConexionPool;
import com.aerolinea.entidad.Vuelo;
import com.aerolinea.operaciones.Operaciones;
import com.aerolinea.utilerias.Tabla;
import java.io.IOException;
```

```
import java.sql.SQLException;
import java.sql.Timestamp;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name = "Vuelos", urlPatterns = {"/Vuelos"})
public class Vuelos extends HttpServlet {

   @Override
   protected void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      String accion = request.getParameter("accion");
      if(accion == null) {
         if(request.getSession().getAttribute("resultado")!=null) {
            request.setAttribute("resultado", request.getSession().getAttribute("resultado"));
            request.getSession().removeAttribute("resultado");
         }
         try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();

            String sql = "select idvuelo, FORMAT(fecha, 'dd/MM/yyyy HH:mm', 'en-US') as date, \n" +
                  "ori.aeropuerto as origen,\n" +
                  "dest.aeropuerto as destino,\n" +
                  "av.descripcion as avion,\n" +
                  "av.capacidad as capacidad,\n" +
                  "vuelo.estado as estado\n" +
                  "from vuelo\n" +
                  "inner join aeropuerto as ori on ori.idaeropuerto = vuelo.idorigen\n" +
                  "inner join aeropuerto as dest on dest.idaeropuerto = vuelo.iddestino\n" +
```

```
        "inner join avion as av on av.idavion = vuelo.idavion";
String[][] vuelos=null;
if (request.getParameter("idorigen")!=null){
   List<Object> param = new ArrayList();
   param.add(request.getParameter("idorigen"));
   request.setAttribute("idorigen", request.getParameter("idorigen"));
   request.setAttribute("origen", request.getParameter("origen"));
   sql+=" where vuelo.idorigen = ?";
   vuelos= Operaciones.consultar(sql, param);
}
//declaracion de cabeceras a usar en la tabla HTML
String[] cabeceras = new String[]{
 "Id Vuelo",
 "Fecha y Hora",
 "Origen",
 "Destino",
 "Avión",
 "Capacidad",
 "Estado"
};
//variable de tipo Tabla para generar la Tabla HTML
Tabla tab = new Tabla(vuelos, //array que contiene los datos
"100%", //ancho de la tabla px | %
Tabla.STYLE.TABLE01, //estilo de la tabla
Tabla.ALIGN.CENTER,  // alineacion de la tabla
cabeceras); //array con las cabeceras de la tabla
//boton eliminar
tab.setEliminable(true);
//url del proyecto
tab.setPageContext(request.getContextPath());
tab.setSeleccionable(false);
//pagina encargada de eliminar
tab.setPaginaEliminable("/Vuelos?accion=eliminar");
//pagina encargada de seleccion para operaciones
//tab.setPaginaSeleccionable("/Paises");
//icono para modificar y eliminar
//tab.setIconoModificable("/iconos/edit.png");
tab.setIconoEliminable("/iconos/delete.png");
//columnas seleccionables
//tab.setColumnasSeleccionables(new int[]{1});
//pie de tabla
```

```
            tab.setPie("Resultado Vuelos");
            //imprime la tabla en pantalla
            String tabla01="No hay datos";
            if (vuelos!=null)
               tabla01= tab.getTabla();
            request.setAttribute("tabla", tabla01);
            request.getRequestDispatcher("vuelos/vuelos_consulta.jsp").forward(request, response);
         } catch(Exception ex) {
            System.out.println(ex.getMessage());
            try {
               Operaciones.rollback();
            } catch (SQLException ex1) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
         } finally {
            try {
               Operaciones.cerrarConexion();
            } catch (SQLException ex) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
         }
      } else if(accion.equals("listado_origen")) {
         try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();

            String sql = "select aero.idaeropuerto, aero.aeropuerto,\n" +
                  "pa.pais, aero.ciudad\n" +
                  "from aeropuerto as aero\n" +
                  "inner join pais as pa on pa.idpais = aero.idpais";

            String[][] origenes = Operaciones.consultar(sql, null);

            //declaracion de cabeceras a usar en la tabla HTML
            String[] cabeceras = new String[]{
              "Id Aeropuerto",
              "Aeropuerto",
              "Pais",
              "Ciudad"
```

```
            };
            //variable de tipo Tabla para generar la Tabla HTML
            Tabla tab = new Tabla(origenes, //array que contiene los datos
            "100%", //ancho de la tabla px | %
            Tabla.STYLE.TABLE01, //estilo de la tabla
            Tabla.ALIGN.LEFT,  // alineacion de la tabla
            cabeceras); //array con las cabeceras de la tabla
            tab.setMetodoFilaSeleccionable("_Seleccionar_");
            //url del proyecto
            tab.setPageContext(request.getContextPath());
            tab.setFilaSeleccionable(true);
            //icono para modificar y eliminar
//              tab.setIconoModificable("/iconos/edit.png");
//              tab.setIconoEliminable("/iconos/delete.png");
            //columnas seleccionables
            tab.setColumnasSeleccionables(new int[]{1});
            //pie de tabla
            tab.setPie("Resultado Aeropuertos");
            //imprime la tabla en pantalla
            String tabla01="No hay datos";
            if (origenes!=null)
               tabla01= tab.getTabla();
            request.setAttribute("tabla", tabla01);
            request.getRequestDispatcher("vuelos/aeropuertos_origen.jsp").forward(request,
response);
         } catch(Exception ex) {
            try {
               Operaciones.rollback();
            } catch (SQLException ex1) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
         } finally {
            try {
               Operaciones.cerrarConexion();
            } catch (SQLException ex) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
         }
      } else if(accion.equals("listado_destino")) {
         try {
            Conexion conn = new ConexionPool();
```

```
conn.conectar();
Operaciones.abrirConexion(conn);
Operaciones.iniciarTransaccion();

String sql = "select aero.idaeropuerto, aero.aeropuerto,\n" +
      "pa.pais, aero.ciudad\n" +
      "from aeropuerto as aero\n" +
      "inner join pais as pa on pa.idpais = aero.idpais";

String[][] destinos = Operaciones.consultar(sql, null);

//declaracion de cabeceras a usar en la tabla HTML
String[] cabeceras = new String[]{
 "Id Aeropuerto",
 "Aeropuerto",
 "Pais",
 "Ciudad"
};
//variable de tipo Tabla para generar la Tabla HTML
Tabla tab = new Tabla(destinos, //array que contiene los datos
"100%", //ancho de la tabla px | %
Tabla.STYLE.TABLE01, //estilo de la tabla
Tabla.ALIGN.LEFT,  // alineacion de la tabla
cabeceras); //array con las cabeceras de la tabla
//url del proyecto
tab.setPageContext(request.getContextPath());
tab.setFilaSeleccionable(true);
tab.setMetodoFilaSeleccionable("_Seleccionar_");
//icono para modificar y eliminar
tab.setIconoModificable("/iconos/edit.png");
tab.setIconoEliminable("/iconos/delete.png");
//columnas seleccionables
tab.setColumnasSeleccionables(new int[]{1});
//pie de tabla
tab.setPie("Resultado Aeropuertos");
//imprime la tabla en pantalla
String tabla01="No hay datos";
if (destinos!=null)
   tabla01= tab.getTabla();
request.setAttribute("tabla", tabla01);
request.setAttribute("tabla", tabla01);
```

```
            request.getRequestDispatcher("vuelos/aeropuertos_destino.jsp").forward(request,
response);
        } catch(Exception ex) {
            try {
                Operaciones.rollback();
            } catch (SQLException ex1) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
        } finally {
            try {
                Operaciones.cerrarConexion();
            } catch (SQLException ex) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    } else if(accion.equals("listado_avion")) {
        try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();

            String sql = "select idavion, descripcion, capacidad from avion";

            String[][] aviones = Operaciones.consultar(sql, null);

            //declaracion de cabeceras a usar en la tabla HTML
            String[] cabeceras = new String[]{
              "Id Avion",
              "Avion",
              "Capacidad"
            };
            //variable de tipo Tabla para generar la Tabla HTML
            Tabla tab = new Tabla(aviones, //array que contiene los datos
            "100%", //ancho de la tabla px | %
            Tabla.STYLE.TABLE01, //estilo de la tabla
            Tabla.ALIGN.LEFT,  // alineacion de la tabla
            cabeceras); //array con las cabeceras de la tabla
            //url del proyecto
            tab.setPageContext(request.getContextPath());
            tab.setFilaSeleccionable(true);
```

```
            tab.setMetodoFilaSeleccionable("_Seleccionar_");
            //icono para modificar y eliminar
            tab.setIconoModificable("/iconos/edit.png");
            tab.setIconoEliminable("/iconos/delete.png");
            //columnas seleccionables
            tab.setColumnasSeleccionables(new int[]{1});
            //pie de tabla
            tab.setPie("Resultado Aeropuertos");
            //imprime la tabla en pantalla
            String tabla01="No hay datos";
            if (aviones!=null)
               tabla01= tab.getTabla();
            request.setAttribute("tabla", tabla01);
            request.getRequestDispatcher("vuelos/listado_aviones.jsp").forward(request, response);
         } catch(Exception ex) {
            try {
               Operaciones.rollback();
            } catch (SQLException ex1) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
         } finally {
            try {
               Operaciones.cerrarConexion();
            } catch (SQLException ex) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
         }
      } else if(accion.equals("eliminar")) {
         try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();
            Vuelo p = Operaciones.eliminar(Integer.parseInt(request.getParameter("id")), new Vuelo());
            if(p.getIdvuelo()!=0) {
               request.getSession().setAttribute("resultado", 1);
            } else {
               request.getSession().setAttribute("resultado", 0);
            }
            Operaciones.commit();
         } catch(Exception ex) {
```

```
            try {
                Operaciones.rollback();
            } catch (SQLException ex1) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
            request.getSession().setAttribute("resultado", 0);
        } finally {
            try {
                Operaciones.cerrarConexion();
            } catch (SQLException ex) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        response.sendRedirect(request.getContextPath()+"/Vuelos");
    }
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    String idorigen = request.getParameter("txtIdorigen");
    String origen = request.getParameter("txtOrigen");
    String fecha = request.getParameter("txtFecha");
    String idavion = request.getParameter("txtIdavion");
    String iddestino = request.getParameter("txtIddestino");
    try {
        Conexion conn = new ConexionPool();
        conn.conectar();
        Operaciones.abrirConexion(conn);
        Operaciones.iniciarTransaccion();
        Vuelo vuelo = new Vuelo();
        vuelo.setIdavion(Integer.parseInt(idavion));
        vuelo.setIdorigen(Integer.parseInt(idorigen));
        vuelo.setIddestino(Integer.parseInt(iddestino));
        vuelo.setEstado("Disponible");
        Date date = new SimpleDateFormat("dd/MM/yyyy HH:mm").parse(fecha);
        vuelo.setFecha(new Timestamp(date.getTime()));
        vuelo = Operaciones.insertar(vuelo);
        if(vuelo.getIdvuelo()!=0) {
            request.getSession().setAttribute("resultado", 1);
        } else {
```

```
            request.getSession().setAttribute("resultado", 0);
        }
        Operaciones.commit();

response.sendRedirect(request.getContextPath()+"/Vuelos?idorigen="+idorigen+"&origen="+origen);
    } catch(Exception ex) {
        try {
            Operaciones.rollback();
        } catch (SQLException ex1) {
            Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
        }
        request.getSession().setAttribute("resultado", 2);
        ex.printStackTrace();
    } finally {
        try {
            Operaciones.cerrarConexion();
        } catch (SQLException ex) {
            Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
  }
}
```

III- Tarea

- ✓ Realizar las validaciones correspondientes al ejemplo de la guía, cajas de texto vacías, origen diferente de destino, de fecha, etc.
- ✓ Agregar el precio a la inserción y consulta de vuelos.
- ✓ Crear el Maestro-Detalle para Reservaciones, tomar en cuenta que debe haber un área para el registro previo del usuario que desea hacer la reservación, siendo este un cliente del sitio y por ende debe estar registrado para poder hacer una reservación