**ACCESO A BASES DE DATOS DESDE JAVA (cont):**

Al estudiar SQL, vimos que existían dos tipos de instrucciones.

* **Consultas de selección** (SELECT)

Estas consultas permiten extraer datos de la base de datos. Dicho de otro modo, permiten visualizar información de la base de datos que cumpla un criterio.

Estas consultas no afectan a la base de datos, simplemente muestran información sobre la propia base de datos.

* **Consultas de acción.** (INSERT, DELETE, UPDATE)

Estas consultas realizan una acción sobre la base de datos. Esta acción puede ser:

* + Insertar un nuevo registro en una tabla (INSERT)
  + Borrar un registro o registros de una tabla (DELETE)
  + Modificar los datos de un registro o registros de la tabla (UPDATE)

**Ejecución de consultas de selección y de consultas de acción.**

Hasta ahora se han realizado programas java que ejecutaban consultas de selección sobre la base de datos.

Recuerda que para ejecutar estas consultas se usa el método *executeQuery* del objeto *sentencia* y el resultado de la consulta se almacena en un objeto *ResultSet*:

ResultSet r = sentencia.executeQuery(“select . . .”);

En esta hoja guiada veremos la ejecución de consultas de acción sobre la base de datos desde la aplicación java. Este tipo de consultas se ejecutan usando el método *executeUpdate* del objeto *sentencia*, y no devuelven un resultado concreto, ya que simplemente actúan sobre la base de datos modificando de alguna manera su contenido.

Así pues, para realizar un alta en la base de datos se usará:

sentencia.executeUpdate(“insert . . .”);

Para realizar una modificación en la base de datos se usará:

sentencia.executeUpdate(“update . . . “);

Para realizar una eliminación en la base de datos se usará:

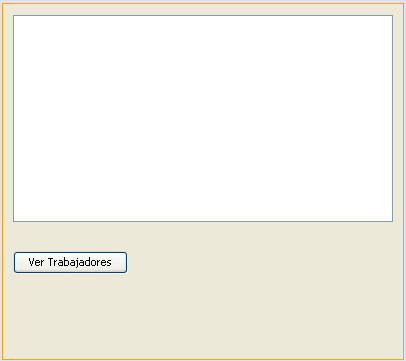
sentencia.executeUpdate(“delete . . . “);

**Cuando la consulta es de acción, las instrucciones SQL no devuelven ningún *ResultSet*, ya que no extraen datos de las tablas, sino que modifican el contenido de éstas.**

**EJERCICIO 1:**

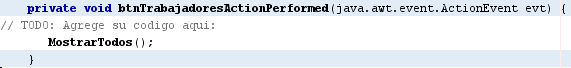
Se quiere realizar una aplicación de base de datos que manipule los datos de los trabajadores de la base de datos MANEMPSA. Esta aplicación permitirá ver el listado de trabajadores y además permitirá introducir nuevos trabajadores.

1. Crea un nuevo proyecto , dentro de este proyecto crea un paquete principal llamado *paqueteprincipal* y dentro de él un JFrame llamado *ventanaprincipal*. Añade a la ventana un JTextPane y un botón:



El botón se llamará ***btnTrabajadores***y el JTextPane se llamará ***panelTexto***.

1. Programa el botón ***btnTrabajadores****.* Se pretende que al pulsar este botón aparezca en el panel *panelTexto* el contenido de la tabla *trabajadores*.



El método ***MostrarTodos***.se encargará de mostrar todos los trabajadores en el *panelTexto*. Programa el método *MostrarTodos* de la siguiente forma:



Básicamente lo que hace es ejecutar una consulta SQL que recoge todos los datos de la tabla trabajadores y luego muestra dichos datos en el panel de la ventana.

Se muestra el listado procurando que las fechas aparezcan con el formato dia-mes-año, que los sueldos aparezcan con la coma decimal y que si el campo matrícula fuera nulo o la cadena vacía “”, entonces aparezca el texto “sin coche”.

1. Añadir los siguientes elementos a la ventana principal:

Panel: *panelAlta*

Cuadros de Texto:

*txtDNI*

*txtNombre*

*txtApellidos*

*txtSueldo*

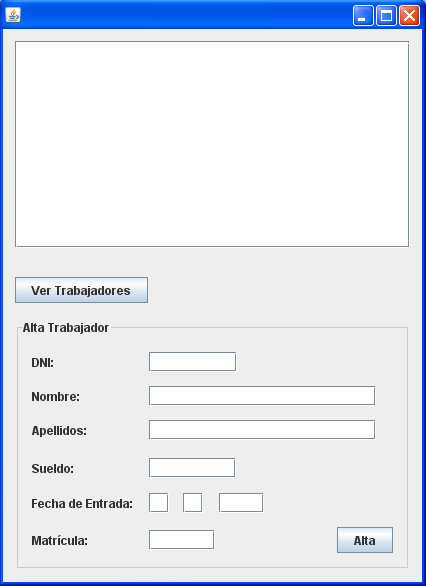
*txtDia*

*txtMes*

*txtAnio*

*txtMatricula*

Botón: *btnAlta*

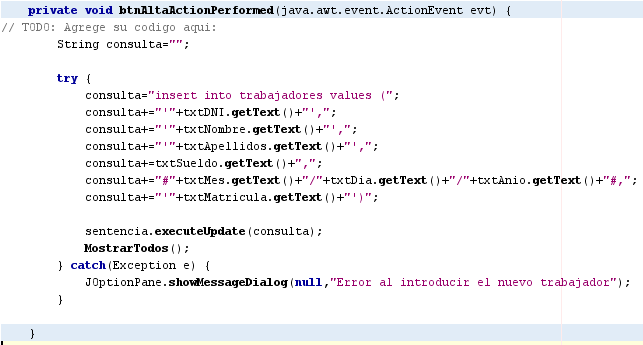


El objetivo de estos elementos es el siguiente:

El usuario introducirá los datos de un nuevo trabajador en las casillas indicadas.

Luego, al pulsar el botón de *Alta*, se introducirá en la tabla *trabajadores* los datos del nuevo trabajador y aparecerá en el panel la lista actualizada de trabajadores incluyendo al nuevo que se ha añadido.

Programar en el botón *Alta* lo siguiente:



Por ejemplo, supongamos que introducimos los siguientes valores en los cuadros de texto:

txtDNI 🡪 11.111.111-A

txtNombre 🡪 María

txtApellidos 🡪 Ruiz

txtSueldo 🡪 1100

txtDia 🡪 10

txtMes 🡪 4

txtAnio 🡪 2001

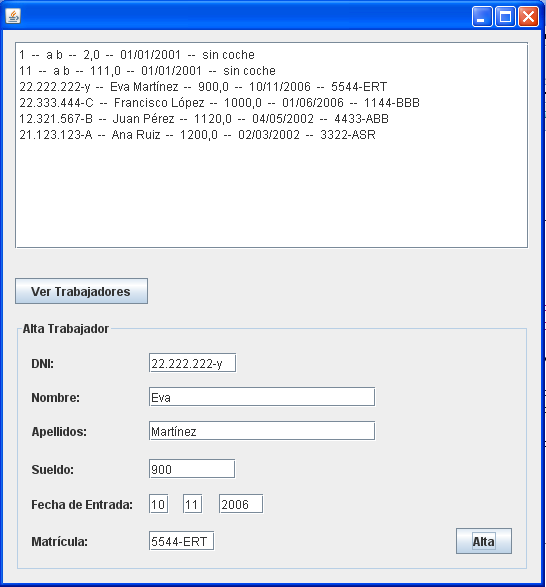
txtMatricula 🡪 4433RET

La concatenación en la variable *consulta* resultaría lo siguiente (en azul los valores de los cuadros de texto, en rosa las cadenas que se concatenan):

**insert into trabajadores values (‘11.111.111-A’,’María’,’Ruiz’,1100,#4/10/2001#,’4433RET’)**

Introduce datos y pulsa el botón Alta.

En la parte superior aparecerá el listado completo incluyendo al nuevo trabajador introducido.



**CONCLUSIÓN**

**Se pueden ejecutar instrucciones SQL del tipo INSERT INTO (insertar registros) usando el objeto sentencia.**

**Para este tipo de instrucciones hay que usar el método *executeUpdate*.**

**Normalmente, será necesario construir una cadena de consulta a través de la concatenación de subcadenas y datos introducidos por el usuario en cuadros de texto.**

**Este tipo de instrucciones SQL no devuelven ningún *ResultSet*, ya que no extraen datos de las tablas, sino que modifican el contenido de éstas.**

**Consultas de Acción. Bajas.**

Las consultas SQL de acción del tipo DELETE son usadas para eliminar registros de las tablas de la base de datos.

Al igual que ocurre con las consultas INSERT INTO, las consultas DELETE haya que ejecutarlas usando el método *executeUpdate* del objeto *sentencia*.

sentencia.executeUpdate(“delete . . . “);

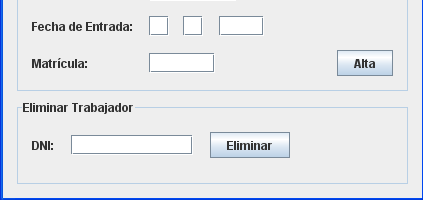
Al ejecutar estas consultas se cambiará el contenido de las tablas de la base de datos y no se devolverá ningún *ResultSet*.

**EJERCICIO 2:**

Mejorar el proyecto del ejercicio 1 de forma que no solo admita la inserción de nuevos trabajadores en la tabla trabajadores de la base de datos MANEMPSA, sino que también permite eliminar a trabajadores de la tabla.

1. Abrir el proyecto ***GestionTrabajadores****.*

Añadir en la parte inferior de la ventana un nuevo panel:



Panel: *panelBajas*

Cuadro de texto: *txtDNIEliminar*

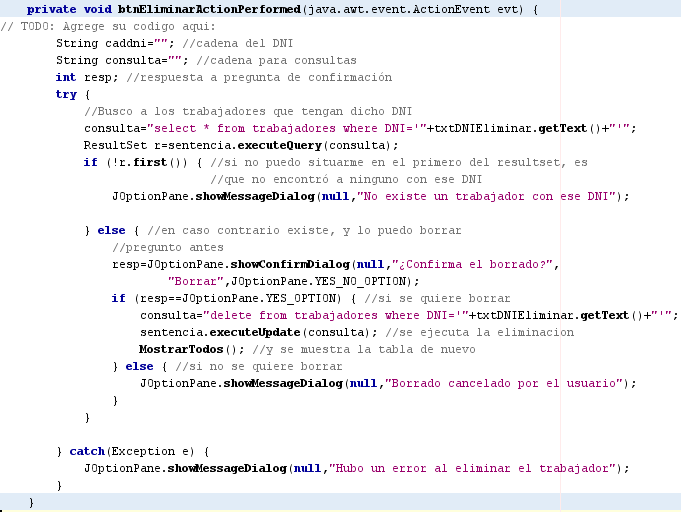
Botón: *btnEliminar*

La mejora que se pretende hacer es la siguiente:

* + El usuario introducirá en el cuadro de texto *txtDNIEliminar* el DNI del trabajador que quiere eliminar.
  + Al pulsar el botón *Eliminar* el trabajador con dicho DNI se eliminará de la tabla.
  + Antes de eliminar al trabajador, se le pedirá al usuario una confirmación, para evitar un borrado accidental.

Si el DNI introducido no se correspondiera con ningún trabajador de la tabla, entonces se mostrará un mensaje indicándolo.

1. Para hacer esto, en el *actionPerformed* del botón *Eliminar* se programa lo siguiente:



Lo primero que se hace es una consulta de selección (SELECT) usando la típica concatenación de cadenas:

consulta="select \* from trabajadores where DNI='"+txtDNIEliminar.getText()+"'";

ResultSet r=sentencia.executeQuery(consulta);

Esto se hace para extraer aquellos trabajadores que tengan el DNI introducido en el cuadro de texto *txtDNIEliminar*. La idea es saber si existe en la tabla algún trabajador con dicho DNI. Al ejecutar esta consulta, el *ResultSet* r se llenará con trabajadores que tengan ese DNI.

A continuación viene:

if (!r.first()) {

Este if significa: “si no puedo colocarme en el primer elemento del *ResultSet* r”.

Si no es posible colocarse en el primer elemento del resultado de la consulta será porque no hay un primer elemento, o dicho de otra forma, porque el *ResultSet* está vacío, o dicho de otra forma más, porque no se encontró a nadie que tuviera el DNI indicado.

Si esto fuera así, no existe un trabajador con dicho DNI y por tanto no puede ser borrado. Así pues se muestra un mensaje con un JOptionPane indicando dicha circunstancia.

En caso contrario, existe ese trabajador y por tanto podemos borrarlo.

resp=JOptionPane.showConfirmDialog(null,"¿Confirma el borrado?",

"Borrar",JOptionPane.YES\_NO\_OPTION);

if (resp==JOptionPane.YES\_OPTION) {

Este trozo de código le pregunta al usuario si quiere borrar realmente al trabajador. Si la respuesta es sí (YES\_OPTION) se efectuará el borrado tal como se indica a continuación:

consulta="delete from trabajadores where DNI='"+txtDNIEliminar.getText()+"'";

sentencia.executeUpdate(consulta); //se ejecuta la eliminacion

MostrarTodos(); //y se muestra la tabla de nuevo

**CONCLUSIÓN**

**Se pueden ejecutar instrucciones SQL del tipo DELETE (borrar registros) usando el objeto sentencia.**

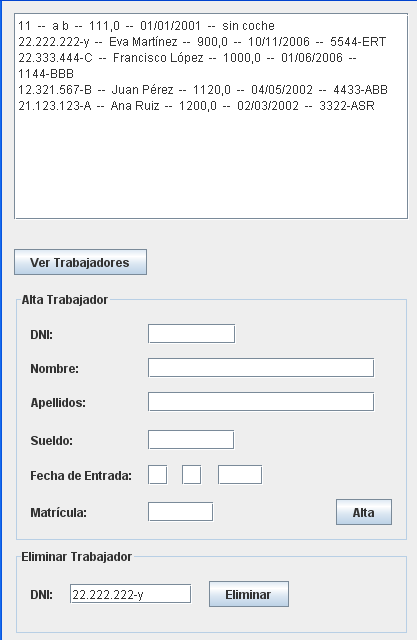
**Para este tipo de instrucciones hay que usar el método *executeUpdate*.**

**Normalmente, será necesario construir una cadena de consulta DELETE a través de la concatenación de subcadenas y datos introducidos por el usuario en cuadros de texto.**

**Este tipo de instrucciones SQL no devuelven ningún *ResultSet*, ya que no extraen datos de las tablas, sino que modifican el contenido de éstas.**

**Para realizar borrados será necesario habitualmente el comprobar que los registros a borrar existen en la tabla. Esto se hará ejecutando una consulta SELECT y comprobando si el *ResultSet* resultante contiene algún registro, usando el método *first*.**

**También se recomienda pedir confirmación al usuario antes de realizar borrados en las tablas.**



Introduzca el DNI del trabajador que quiere borrar y luego pulse *Eliminar*.

Al hacerlo el listado se actualizará y ya no mostrará el trabajador con dicho DNI ya que habrá sido borrado.

Si el DNI no existiera el borrado no se produciría y mostraría un mensaje de error.

**EJERCICIO 3:**

**Consultas de Acción. Modificaciones.**

Las consultas SQL de acción UPDATE (permiten modificar los registros de las tablas)

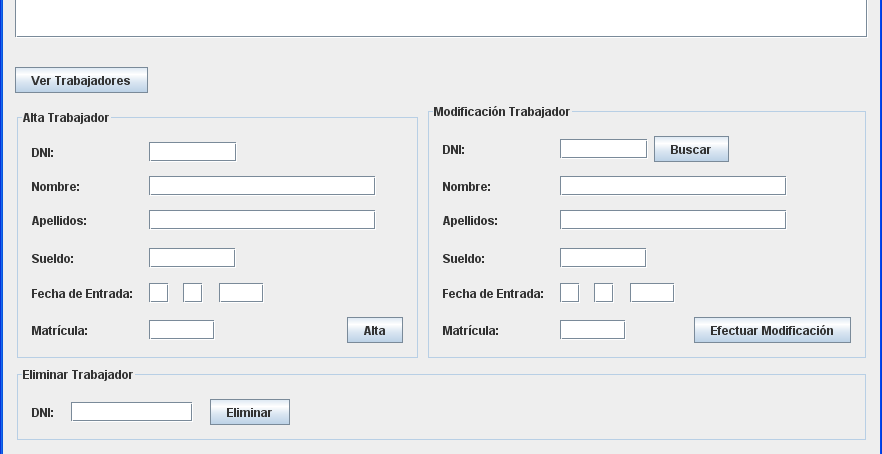
Estas consultas SQL se ejecutan desde el programa a través del método ***executeUpdate*** propio del objeto *sentencia*.

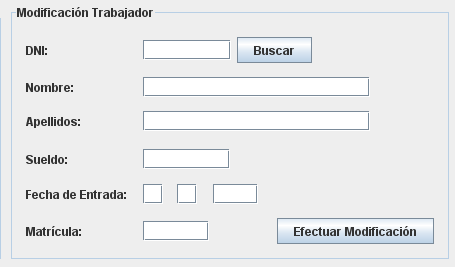
Estas instrucciones SQL afectan al contenido de la base de datos y no devuelven ningún resultado. Es decir, no extraen datos, y por tanto no devuelven un *ResultSet*.

1. En este ejercicio se mejorará el proyecto anterior de forma que no sólo permita añadir y borrar trabajadores de la tabla *trabajadores* de la base de datos MANEMPSA, sino que también permita realizar modificaciones en los datos de los trabajadores de dicha tabla.

Modifica la ventana principal **añadiendo un nuevo panel** con los siguientes componentes:

Nuevo panel





El panel se llamará *panelModificar*

El botón es *btnBuscar*

Los cuadros de texto se llaman respectivamente:

*txtModDNI*

*txtModNombre*

*txtModApellidos*

*txtModSueldo*

*txtModDia*

*txtModMes*

*txtModAnio*

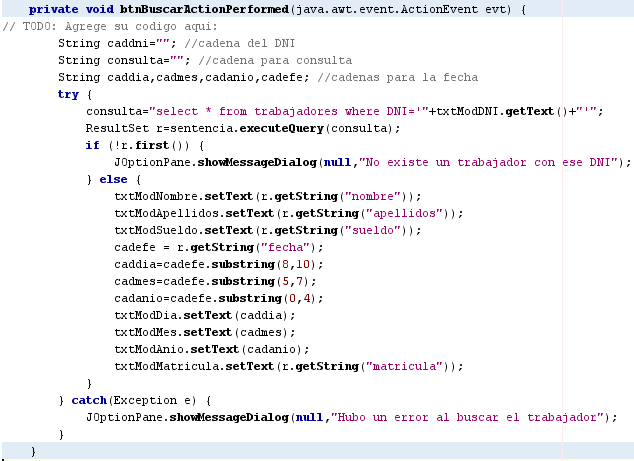
*txtModMatricula*

El botón es *btnModificar*

La mejora que se pretende hacer es la siguiente:

* + El usuario introducirá en el cuadro de texto *txtModDNI* el DNI del trabajador cuyos datos se quieren modificar.
  + Al pulsar el botón *Buscar*, el resto de cuadros de texto se rellenarán con los datos del trabajador al que pertenece dicho DNI. (Si dicho DNI no existiera, se mostraría un mensaje de error)
  + Cuando aparezcan los datos del trabajador en los cuadros de texto, el usuario podrá realizar las modificaciones pertinentes.
  + Al pulsarse el botón *Efectuar Modificación* se modificarán los datos en la tabla.

1. Primero **programa el botón de búsqueda.** Entre en el *actionPerformed* de este botón y programa lo siguiente:



Lo primero que hace el código es construir una consulta de selección que busque a todos los trabajadores que tengan el DNI introducido en el cuadro de texto *txtModDNI*.

Ejecuta la consulta y almacena el resultado en un típico *ResultSet*. Luego comprueba a través del siguiente if si no existe un trabajador con dicho DNI en el *ResultSet*:

if (!r.first()) {

En ese caso muestra un mensaje de error indicando que no se ha encontrado dicho trabajador.

En caso contrario muestra los datos del trabajador encontrado en los cuadros de texto. Para ello, se extrae del *ResultSet* cada campo y se coloca en el cuadro de texto correspondiente. Por ejemplo:

txtModNombre.setText(r.getText(“nombre”));

Recoge el valor del campo *nombre*, y lo sitúa en el cuadro de texto *txtModNombre*.

En el caso de la fecha de entrada del trabajador, se extrae la fecha y luego se extrae de la cadena el día, el mes y el año, situando cada dato en su cuadro de texto correspondiente.

cadefe = r.getString("fecha");

caddia=cadefe.substring(8,10);

cadmes=cadefe.substring(5,7);

cadanio=cadefe.substring(0,4);

txtModDia.setText(caddia);

txtModMes.setText(cadmes);

txtModAnio.setText(cadanio);

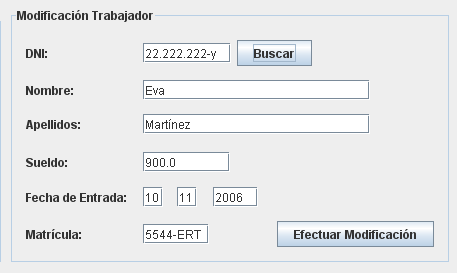
Recuerda que cuando se extrae una fecha de un *ResultSet*, su formato es:

año-mes-dia-hora:minutos

Por ejemplo: 2001-12-23-00:00

Así pues, si el DNI introducido pertenece a uno de los trabajadores de la tabla, sus datos aparecerán en los cuadros de texto cuando se pulse el botón *Buscar*.

Para ejecutar el programa se escribe un DNI en el cuadro del DNI de búsqueda y pulsa *Buscar*.

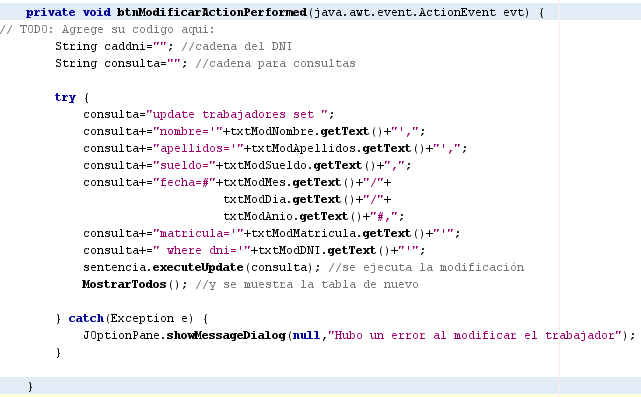


Se introduce un DNI y luego se pulsa *Buscar*.

Si existe un trabajador con ese DNI, sus datos aparecerán en los cuadros de texto.

Cuando aparezcan los datos del trabajador en los cuadros de texto el usuario los modificará según le interese. Luego, al pulsar el botón *Efectuar Modificación* los nuevos datos serán actualizados en la tabla.

1. Ahora **programa el botón de *Efectuar Modificación****.* Entre en el *actionPerformed* de este botón y programa lo siguiente:



Lo primero que se hace en este código es crear una consulta de actualización (UPDATE) a través de una concatenación de cadenas.

Si observas la concatenación, verá que el resultado es una instrucción SQL de tipo UPDATE en la que los datos que se actualizan son precisamente los datos introducidos en los cuadros de texto del panel de modificación.

Por ejemplo, si estos fueran los valores de los cuadros de texto del panel de modificación:

txtModDNI 🡪 11.111.111-A

txtModNombre 🡪 Juan

txtModApellidos 🡪 Perez

txtModSueldo 🡪 1100

txtModDia 🡪 4

txtModMes 🡪 12

txtModAnio 🡪 2001

txtModMatricula 🡪 1234-ABC

La instrucción ya construida por concatenación sería la siguiente:

update trabajadores

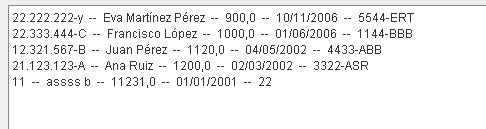
set nombre=’Juan’,apellidos=’Perez’,sueldo=1100,fecha=#12/4/2001#, matricula=’1234-ABC’

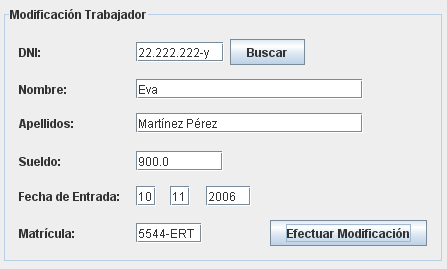
where dni=’11.111.111-A’

Esta instrucción SQL de tipo UPDATE incluye una cláusula WHERE con una condición de forma que solo se modifica aquel trabajador que tenga el DNI introducido en el cuadro de texto *txtModDNI*.

Una vez construida la consulta UPDATE, esta se ejecuta usando el objeto *sentencia* y luego se muestra el contenido de la tabla trabajadores en el panel a través de una llamada al método *MostrarTodos*.

Todo este código está rodeado de un try...catch para evitar errores inesperados.





Se introduce un DNI y se le da a *Buscar*.

Los cuadros de texto se rellenan con los datos del trabajador.

Se realiza la modificación que se quiera en los cuadros de texto.

Finalmente se pulsa *Efectuar Modificación* para que se realice la modificación.

En el panel aparecerá la lista actualizada de los trabajadores.

**CONCLUSIÓN**

**Se pueden ejecutar instrucciones SQL del tipo UPDATE (modificar registros) usando el objeto sentencia.**

**Para este tipo de instrucciones hay que usar el método *executeUpdate*.**

**Normalmente, será necesario construir una cadena de consulta UPDATE a través de la concatenación de subcadenas y datos introducidos por el usuario en cuadros de texto.**

**Este tipo de instrucciones SQL no devuelven ningún *ResultSet*, ya que no extraen datos de las tablas, sino que modifican el contenido de éstas.**

**Es una buena idea dar alguna opción de búsqueda que permita encontrar al registro a modificar y que sus datos aparezcan en varios cuadros de texto, de forma que el usuario tenga más facilidad a la hora de realizar los datos.**