

UNIVERSIDAD CATÓLICA DE SANTA MARÍA
PROGRAMA PROFESIONAL DE INGENIERÍA DE SISTEMAS

SESIÓN N° 06:

Bases de Datos

I

OBJETIVOS

- ❖ Conocer las características del almacenamiento en bases de datos.
- ❖ Conocer la arquitectura de una base de datos y la forma como se gestionan las bases de datos de manera básica.
- ❖ Aprender a conectar base de datos con aplicaciones web.
- ❖ Aprender a diseñar aplicaciones web con conexión a bases de datos.
- ❖ Aprender a configurar y usar controles utilizados para trabajar con bases de datos

II

TEMAS A TRATAR

- ❖ Introducción.
- ❖ ADO.NET
- ❖ Los Proveedores De Acceso a Datos
- ❖ Structured Query Language
- ❖ Resumen

III

MARCO TEORICO

1. INTRODUCCIÓN

El mundo moderno produce mucha información dicha información está disponible en muchos lugares, uno de esos lugares es internet, a través del tiempo las aplicaciones se han utilizado para poder almacenar información, es decir que han servido de interfaces para la captura ordenada y organizada de datos que finalmente van a ser depositados en una base de datos a través de diferentes mecanismos y tecnologías. la utilidad de una aplicación radica en la posibilidad de procesar datos, realizando operaciones que permiten crear nuevas fuentes de información a partir de los datos inicialmente ingresados.

las aplicaciones web poseen la capacidad de capturar datos de diferente tipo, el poder almacenar la información capturada es un factor clave en el proceso computacional, los datos no solamente se almacenan para preservar la información que representan, los datos deben ser preservados para realizar nuevas operaciones en función de los requerimientos de los usuarios de dicha información. tener la capacidad de preservar los datos es muy importante para cualquier aplicación de computación más aún si esta aplicación se ejecuta sobre internet, es por eso que nos resulta un factor muy importante e imprescindible tener la capacidad de almacenar datos y poder disponer de los mismos para su procesamiento y así producir la información que permitirá a los usuarios tomar decisiones informadas en el momento en que tengan que realizar procesos

haciendo uso de la información.

la arquitectura de internet permite tener un servidor de base de datos en cuál almacena toda la información que dispone una aplicación web para su funcionamiento. la interacción de los usuarios con las aplicaciones web nos llevan a hacer consultas, las cuales son posibles a través de la extracción de datos que están registrados en bases de datos de diferentes tecnologías, las páginas web al final resultan como interfaces que presentan la información que las aplicaciones extraen de las bases de datos, este tipo de modelo computacional está extendido y nos permite hacer un uso eficiente y eficaz de la información a través de las consultas que se hacen desde internet.

2. ADO.NET

Es una estrategia de un modelo de objetos para acceder a los datos que ofrece Microsoft a 223 es un modelo ampliado que comprende necesidades que el modelo anterior no podía ofrecer está diseñado para trabajar con conjuntos de datos desconectados lo que permite reducir el tráfico de red utilizan el modelo del lenguaje XML como formato universal de transmisión de datos.

Dentro del espacio de nombres System.Data encontramos las clases compartidas que constituyen el eje central de ADO: NET, y son las siguientes:

DataSet: almacén de datos por excelencia en ADO.NET. Representa una base de datos desconectada del proveedor de datos. Almacena tablas y sus relaciones.

DataTable: un contenedor de datos. Estructurado como un conjunto de filas (DataRow) y columnas (DataColumn).

DataRow: registro que almacena n valores. Representación en ADO .NET de una fila/tupla de una tabla de la base de datos.

DataColumn: contiene la definición de una columna. Metadatos y datos asociados a su dominio.

3. LOS PROVEEDORES DE ACCESO A DATOS

Es la capa inferior de la parte correspondiente de acceso a datos y es la responsable de establecer la comunicación con las fuentes de datos. En este conjunto de nombre de espacios, encontraremos casi siempre las clases

Connection, Command, DataAdapter y DataReader como las clases más generales, las cuales nos permiten establecer la conexión con la fuente de datos.

Proveedores de acceso a datos de .NET Framework

Dentro del entorno .NET Framework, encontraremos un nutrido conjunto de proveedores de acceso a datos.

- ODBC .NET Data Provider
- OLEDB .NET Data Provider
- ORACLE Client .NET Data Provider
- SQL Server .NET Data Provider

Estos proveedores de acceso a datos incluidos en Microsoft .NET Framework, los podemos encontrar en los nombres de espacio.

- Imports System.Data.SqlClient.
- Imports System.Data.OleDb.
- Imports CoreLab.MySql.
- Imports System.Data.Odbc.

- Imports System.Data.Oracle.

El Objeto Connection: Este objeto es el encargado de establecer una conexión física con una Base de Datos determinada.

El Objeto Command: Este objeto es el que representa una determinada sentencia SQL o un Stored Procedure, juntamente con el objeto.

DataAdapter: El Objeto DataAdapter: Este objeto es quizás el objeto más complejo ya la vez complicado de todos los que forman parte de un proveedor de acceso a datos en NET. Cuando se establece una comunicación entre una fuente de datos y un DataSet, se utiliza como intermediario a un objeto DataAdapter. A su vez, un DataAdapter contiene 4 objetos que son:

1. selectCommand: Realiza las acciones a de selección de datos.
2. DeleteCommand: Realiza las acciones de borrados de datos.
3. InsertCommand: Realiza las acciones de inserción de datos.
4. UpdateCommand: Realiza las acciones de actualización de datos.

El objeto DataReader: Es utilizado en una sola dirección de datos. Se trata de un objeto muy rápido. Puede usar a su vez el objeto Command o el método ExecuteReader

a) **CONCEPTO DATABINDING**

Databinding es una expresión de enlace de datos. Es una forma rápida y sencilla de manejar la fuente de datos mediante su enlace con controles(TextBox) o clases.

4. **STRUCTURED QUERY LANGUAGE**

SQL (por sus siglas en inglés Structured Query Language; en español lenguaje de consulta estructurada) es un lenguaje de dominio específico utilizado en programación, diseñado para administrar, y recuperar información de sistemas de gestión de bases de datos relacionales.² Una de sus principales características es el manejo del álgebra y el cálculo relacional para efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas.

Originalmente basado en el álgebra relacional y en el cálculo relacional, SQL consiste en un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de control de datos. El alcance de SQL incluye la inserción de datos, consultas, actualizaciones y borrado, la creación y modificación de esquemas y el control de acceso a los datos. También el SQL a veces se describe como un lenguaje declarativo, también incluye elementos procesales.

SQL fue uno de los primeros lenguajes comerciales para el modelo relacional de Edgar Frank Codd como se describió en su artículo de investigación de 1970 El modelo relacional de datos para grandes bancos de datos compartidos. A pesar de no adherirse totalmente al modelo relacional descrito por Codd, pasó a ser el lenguaje de base de datos más usado.

SQL pasó a ser el estándar del Instituto Nacional Estadounidense de Estándares (ANSI) en 1986 y de la Organización Internacional de Normalización (ISO) en 1987. Desde entonces, el estándar ha sido revisado para incluir más características. A pesar de la existencia de ambos estándares, la mayoría de los códigos SQL no son completamente portables entre sistemas de bases de datos diferentes sin otros ajustes.

Los tipos estándar de las sentencias SQL:

- Select para recuperar registros.
- Update para modificar los registros.
- Insert para agregar un nuevo registro.
- Delete para borrar registros.

5. RESUMEN

El trabajo con bases de datos se inicia diseñando una tabla dentro de una base de datos para lo cual hacemos uso del servidor de base de datos SQL Server, la tabla es una estructura organizada para el almacenamiento de datos basado en tuplas, para ello hemos hecho uso de comandos SQL para la creación de la tabla y de la estructura inserta en ella, la estructura responde a un conjunto de campos que tienen de forma muy clara la especificación del tipo de datos que deben almacenar en dicho espacio. posteriormente se han agregado datos a la tabla a través de comandos SQL, también se ha procedido a verificar la forma del almacenamiento haciendo uso de instrucciones SQL, con estas operaciones hemos tenido a disposición una base de datos que contiene una sola tabla de datos debidamente modelada, construida y verificada. posteriormente se ha llevado a cabo un trabajo desconexión de la base de datos desarrollada en SQL y llevada a uso desde una aplicación asp.net, para ello se han utilizado controles los cuales han sido configurados para establecer la conexión con la base de datos y extraer los datos bajo diferentes mecanismos y publicarlos en la interfaz gráfica hecha en nuestra aplicación web. de esta manera se ha conseguido hacer la utilización que los datos almacenados en una base de datos en una aplicación web de una forma bastante sencilla y con mínima programación.

IV

(La práctica tiene una duración de 4 horas)

ACTIVIDADES

1. EXPERIENCIA DE PRÁCTICA N° 01: CREAR UNA BASE DE DATOS

a) CREANDO LA BASE DE DATOS "VENTAS" EN SQL SERVER

1. Ingrese a Inicio/Programas/Microsoft SQL Server /SQL Server Management Studio.
2. Para conectarse al servidor de base de datos SQL Server utilizaremos la cuenta del usuario de Windows por lo que ingresaremos el nombre del servidor (server_name\instance_name) Ej: **DESKTOP-UCV9NOA\SQLEXPRESS**, seleccionamos **Windows Authentication** y presionamos el botón **Connect**.

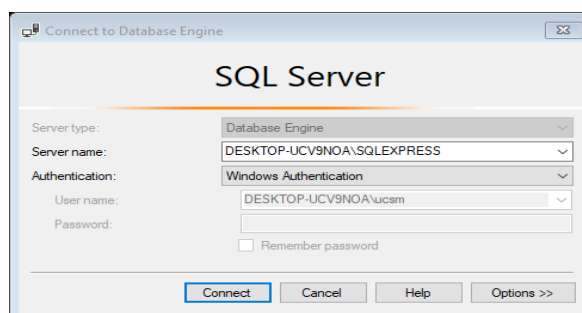


Figura N° 6 - 1: Conexión al servidor de Base de Datos

3. Una vez conectados en el servidor de base de datos, nos ubicamos en el nodo **Databases**, hacemos *clic derecho* y elegimos la opción **New Database...** para crear una Base de Datos a la que llamaremos **VENTAS**, y seleccionamos OK.
4. Con nuestra base de datos creada, seleccionamos el nodo **Tables** ubicado dentro de **VENTAS**, hacemos *clic derecho* y elegimos la opción **New Table...**
5. Vamos a crear una tabla que se llamará **CLIENTES** y que contendrá los siguientes campos:
6. Luego grabamos la tabla creada y le damos por nombre **CLIENTES**. Utilizamos el botón **Save** de la Barra de Herramientas.
7. Notar que una vez que guardamos la tabla, esta aparece en el nodo **Tables** de la base de datos **VENTAS**
8. Para insertar datos en las tablas se utiliza la sentencia **Insert** En la barra de herramientas hacer clic en el botón **New Query**
9. En el panel central copie las siguientes sentencias que permiten ingresar valores a la tabla **CLIENTES**.

```
INSERT INTO CLIENTES VALUES (1, 'Juan Perez', 'Los Portales 310', '959663322', 'jperez@hotmail.com', 30);
INSERT INTO CLIENTES VALUES (2, 'José Díaz', 'Los Angeles de Cayma 250', '959808080', 'jdiaz@hotmail.com', 35);
INSERT INTO CLIENTES VALUES (3, 'Roberto Gonzales', 'Las Orquideas 500', '959757878', 'rgonzales@hotmail.com', 38);
INSERT INTO CLIENTES VALUES (4, 'Ruben Peralta', 'Los Portales 510', '959663322', 'rperalta@hotmail.com', 30);
INSERT INTO CLIENTES VALUES (5, 'María de los Angeles', 'Juan Pablo V. y Guzman L-5', '959272754', 'angeles@hotmail.com', 30);
INSERT INTO CLIENTES VALUES (6, 'Arturo Ordoñez', 'Los Jasmines 250', '959707074', 'aordonez@hotmail.com', 32);
INSERT INTO CLIENTES VALUES (7, 'Josefa Lopez', 'Los Portales 10', '959202023', 'jlopez@hotmail.com', 25);
INSERT INTO CLIENTES VALUES (8, 'Sandra Ramirez', 'Palacio Viejo 100', '958757575', 'sramirez@hotmail.com', 28);
INSERT INTO CLIENTES VALUES (9, 'Pedro Lopez', 'Consuelo 440', '959695656', 'plopez@hotmail.com', 30);
INSERT INTO CLIENTES VALUES (10, 'Juanita Ortega', 'Palacio Viejo 250', '959909088', 'jortega@hotmail.com', 30);
INSERT INTO CLIENTES VALUES (11, 'Marcia Sandoval', 'Pancho Fierro 250', '255000', 'plopez@hotmail.com', 45);
INSERT INTO CLIENTES VALUES (12, 'Manuel Vega', 'Quinta Tristan 520', '425080', 'mvega@hotmail.com', 42);
INSERT INTO CLIENTES VALUES (13, 'Ricardo Salas', 'Coop. Universitaria B-1', '959656522', 'rsalas@hotmail.com', 27);
```

10. Haga clic en el botón **Execute** de la **Barra de Herramientas** y verifique que los registros se inserten con éxito.
11. Para ver el contenido de los datos de la tabla se utiliza la sentencia **SELECT** En el panel consulta escribimos:

```
SELECT * FROM CLIENTES;
```

12. Para ejecutar solo esta sentencia, la seleccionamos y hacemos clic sobre el botón **Execute**. En la parte inferior visualizaremos los registros de la tabla clientes:

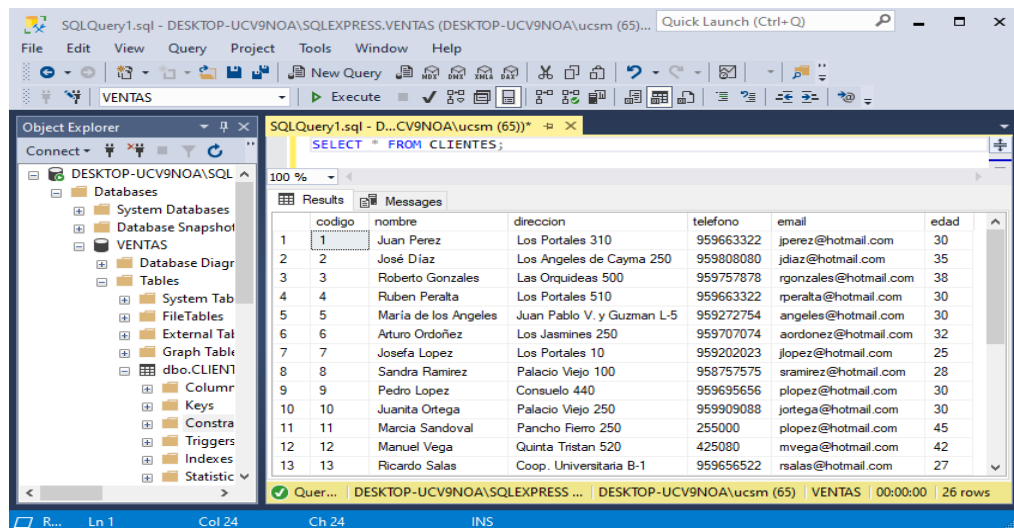


Figura N° 6 - 2: Ejecución de instrucción SELECT.

2. EXPERIENCIA DE PRÁCTICA N° 02: ACCEDER A UNA BASE DE DATOS

a) ACCEDIENDO A UNA BASE DE DATOS DESDE VISUAL STUDIO

1. Crear una aplicación web **ASP.NET Web Forms** vacía nueva y llamar la Base de datos.
2. En la parte superior izquierda de la ventana encontrará el explorador de servidores utilícelo para conectarse al servidor y la base de datos creada en **SQL Server**.

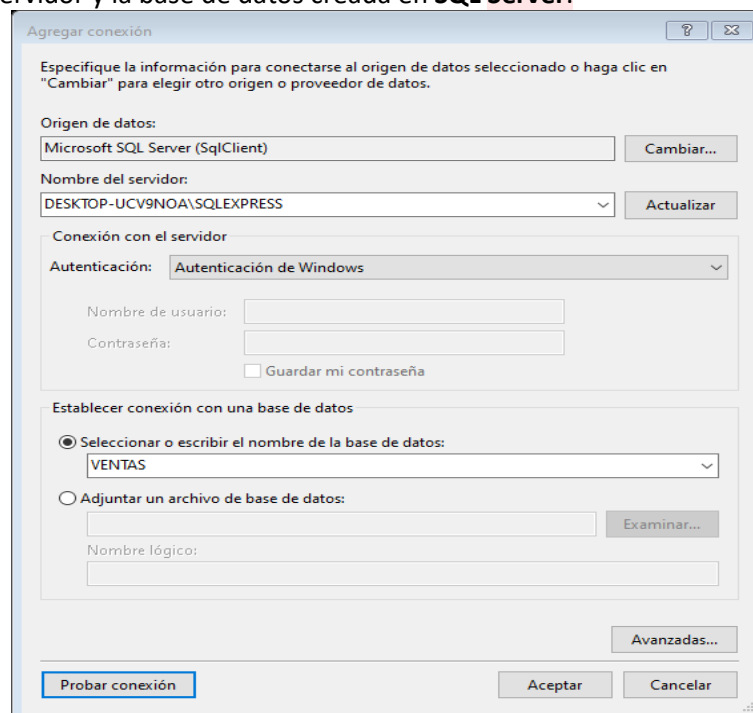


Figura N° 6 - 3: Conexión de la App con la Base de Datos

Utilice el botón **Probar conexión** para verificar la conectividad con la base de datos.

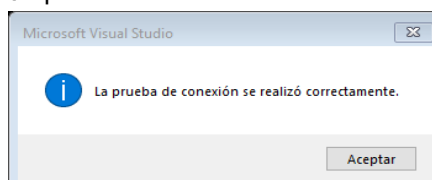


Figura N° 6 - 4: Verificación de conexión.

En el explorador de servidores, elija la división de tablas, luego seleccione la tabla **Clientes** y posteriormente con el botón derecho muestre el menú contextual y seleccione **Mostrar datos de tabla**.

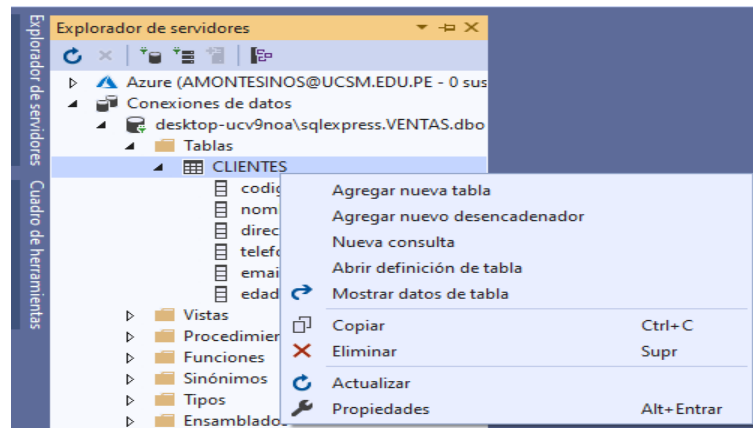


Figura N° 6 - 5: Verificar los datos de la tabla con **Mostrar datos de tabla**.

Se mostrará una ventana con el siguiente resultado:

	codigo	nombre	direccion	telefono	email	edad
1		Juan Perez	Los Portales 310	959663322	jperez@hotmail...	30
2		José Díaz	Los Angeles de ...	959808080	jdiaz@hotmail...	35
3		Roberto Gonzales	Las Orquideas 5...	959757878	rgonzales@hot...	38
4		Ruben Peralta	Los Portales 510	959663322	rperalta@hotm...	30
5		María de los An...	Juan Pablo V. y ...	959272754	angeles@hotm...	30
6		Arturo Ordoñez	Los Jasmines 250	959707074	aordonez@hot...	32
7		Josefa Lopez	Los Portales 10	959202023	jlopez@hotmail...	25
8		Sandra Ramirez	Palacio Viejo 100	958757575	sramirez@hot...	28
9		Pedro Lopez	Consuelo 440	959695656	plopez@hotma...	30
10		Juanita Ortega	Palacio Viejo 250	959909088	jortega@hotma...	30
11		Marcia Sandoval	Pancho Fierro 2...	255000	plopez@hotma...	45
12		Manuel Vega	Quinta Tristan 5...	425080	mvega@hotmail...	42
13		Ricardo Salas	Coop. Universit...	959656522	rsalas@hotmail...	27

Figura N° 6 - 6: Datos de la tabla **Clientes**

b) DISEÑO DEL FORMULARIO PARA BASE DE DATOS

1. Agregar un nuevo **formulario web** en el proyecto.
2. Diseñe un formulario con un control **GridView** de la sección de **Datos** del **Cuadro de Herramientas**:

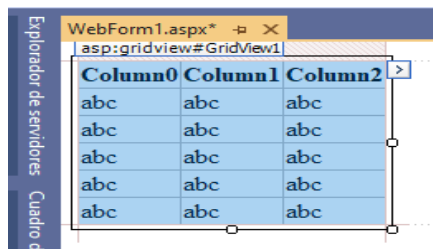


Figura N° 6 - 7: **GridView**

3. Dentro del diálogo **Tareas de GridView** seleccione **Formato automático** para darle un Formato predeterminado (formato de colores) al control **GridView** insertado en el formulario. Elija la opción **Multicolor**.

c) USANDO EL ACCESO DE DATOS DESCONECTADO

1. Presione botón derecho sobre el formulario y elija la opción **Ver código**.

2. Añada como librería el **namespace** luego de todos los **using**.

```
using System.Data.SqlClient;
```

3. Modifique el método **Page_Load** y añada el siguiente código (Cambie el **DataSource** según el nombre de su servidor):

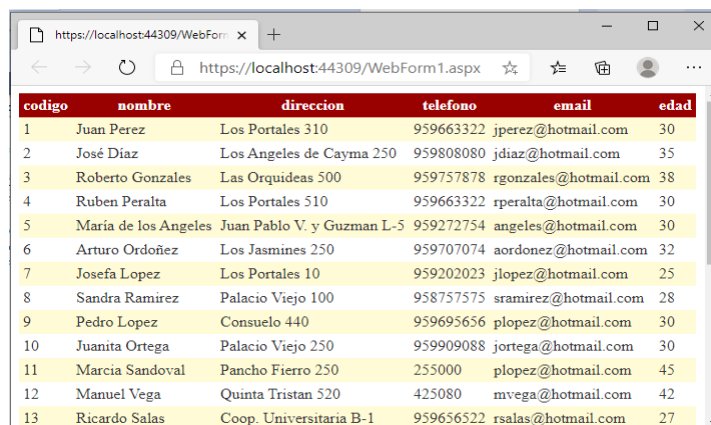
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Data;

namespace basededatos
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            string connectionString = "Data Source = DESKTOP-UCV9NOA\\SQLEXPRESS;" +
                "Initial Catalog = VENTAS; Integrated Security = True";
            string selectSQL = "SELECT * FROM Clientes";
            SqlConnection conexion = new SqlConnection(connectionString);
            SqlCommand comando = new SqlCommand(selectSQL, conexion);
            SqlDataAdapter adapter = new SqlDataAdapter(comando);

            // llenando el dataset
            DataSet ventas = new DataSet();
            adapter.Fill(ventas, "clientes");

            // enlazar el gridview
            GridView1.DataSource = ventas;
            GridView1.DataBind();
        }
    }
}
```

4. Ejecute la aplicación y verifique el funcionamiento, debe aparecer una página web con el contenido siguiente:



The screenshot shows a web browser window with the address bar displaying 'https://localhost:44309/WebForm1.aspx'. The main content area displays a table with 6 columns: 'codigo', 'nombre', 'direccion', 'telefono', 'email', and 'edad'. The table contains 13 rows of client data.

codigo	nombre	direccion	telefono	email	edad
1	Juan Perez	Los Portales 310	959663322	jperez@hotmail.com	30
2	José Díaz	Los Angeles de Cayma 250	959808080	jdiaz@hotmail.com	35
3	Roberto Gonzales	Las Orquideas 500	959757878	rgonzales@hotmail.com	38
4	Ruben Peralta	Los Portales 510	959663322	rperalta@hotmail.com	30
5	María de los Angeles	Juan Pablo V. y Guzman L-5	959272754	angeles@hotmail.com	30
6	Arturo Ordoñez	Los Jasmines 250	959707074	aordonez@hotmail.com	32
7	Josefa Lopez	Los Portales 10	959202023	jlopez@hotmail.com	25
8	Sandra Ramirez	Palacio Viejo 100	958757575	sramirez@hotmail.com	28
9	Pedro Lopez	Consuelo 440	959695656	plopez@hotmail.com	30
10	Juanita Ortega	Palacio Viejo 250	959909088	jortega@hotmail.com	30
11	Marcia Sandoval	Pancho Fierro 250	255000	plopez@hotmail.com	45
12	Manuel Vega	Quinta Tristan 520	425080	mvega@hotmail.com	42
13	Ricardo Salas	Coop. Universitaria B-1	959656522	rsalas@hotmail.com	27

Figura N° 6 - 8: Visualización en la Página Web.

d) USANDO EL CONTROL DE DATOS SQLDATASOURCE

1. Se puede utilizar el control **SqlDataSource** para establecer la conexión con la Base de Datos y ejecutar la consulta anterior, sin necesidad de escribir mucho código. Para probar esto, agregue otro **GridView** en el formulario, y un control **SqlDataSource** de la sección de **Datos** del Cuadro de

Herramientas:

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

SqlDataSource - SqlDataSource1

Figura N° 6 - 9: Controles GridView y SqlDataSource agregados

2. En la vista de código del formulario, modifique para que luzca como lo siguiente:

```
<asp:GridView ID="GridView2" runat="server" DataSourceID="SqlDataSource1" >
</asp:GridView>
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
    SelectCommand ="select * from clientes"
    ConnectionString ="Data Source = DESKTOP-UCV9NOA\SQLEXPRESS;Initial
        Catalog=VENTAS; Integrated Security=True">
</asp:SqlDataSource>
```

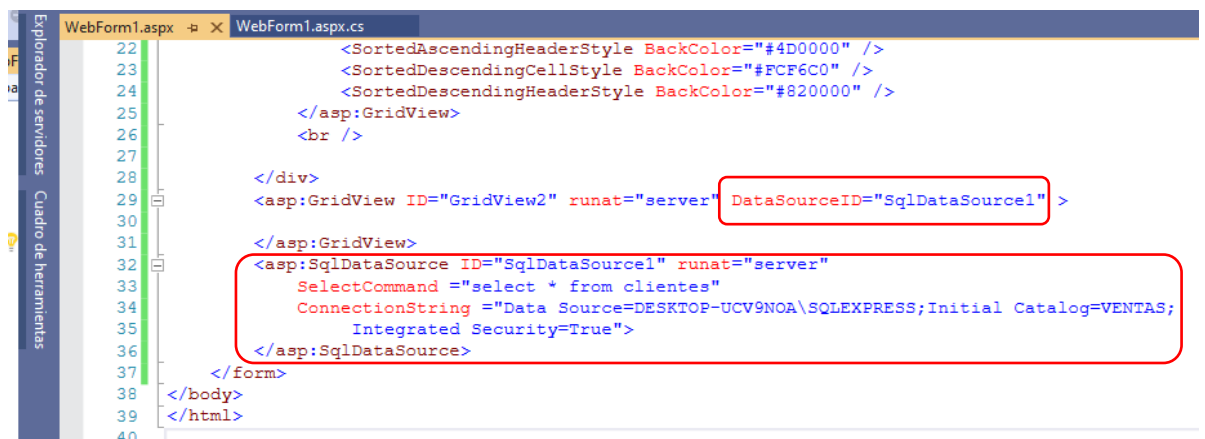


Figura N° 6 - 10: Programación del dataSource

3. Ejecute la aplicación y verifique su funcionamiento, analice las diferencias, ventajas y desventajas entre ambos métodos para recuperar información de una base de datos.
4. Hacer un backup de la base de datos.

3. EXPERIENCIA DE PRÁCTICA N° 03: USANDO ADO.NET

a) RESTAURANDO LA BASE DE DATOS "VENTAS" EN SQL SERVER

1. Ingrese a Inicio/Programas/Microsoft SQL Server/SQL Server Management Studio.
2. Para conectarse al servidor de base de datos SQL Server utilizaremos la cuenta del usuario de Windows por lo que ingresaremos el nombre del servidor (server_name\instance_name) Ej: PC200\SQLEXPRESS, seleccionamos Windows Authentication y presionamos el botón Connect.

3. Una vez conectados vamos a restaurar el **backup** de la base de datos **VENTAS**. Seleccionamos el nodo **Databases**, clic derecho y elegimos la opción **Restore Database...**

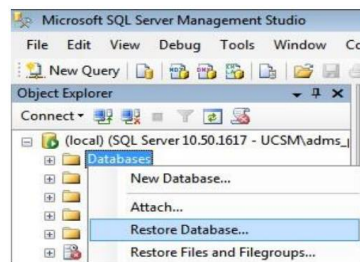


Figura N° 6 - 11: Restaurar Base de Datos en el menú

4. Ingresamos el nombre de la base de datos que vamos a Restaurar, **VENTAS**. Seleccionamos la opción **From device** y hacemos clic sobre el botón (...).

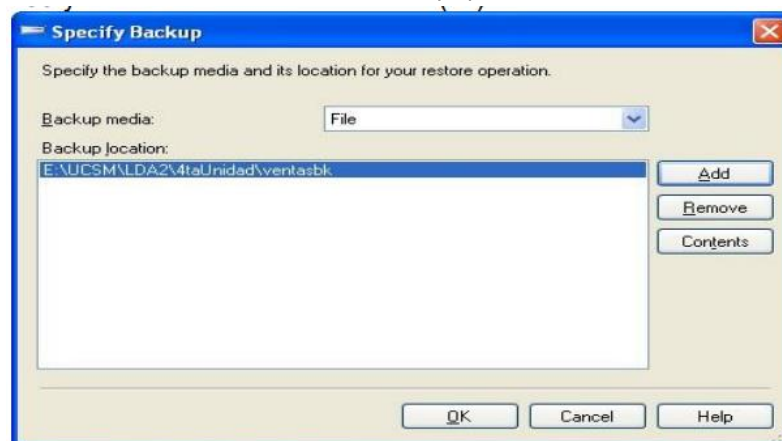


Figura N° 6 - 12: Restaurar Base de Datos elegir Base de Datos

5. En la ventana **Specify Backup**, hacemos clic sobre el botón **Add** y nos aparecerá la siguiente pantalla para seleccionar la ubicación donde tengamos guardado el **backup** de la base de datos:

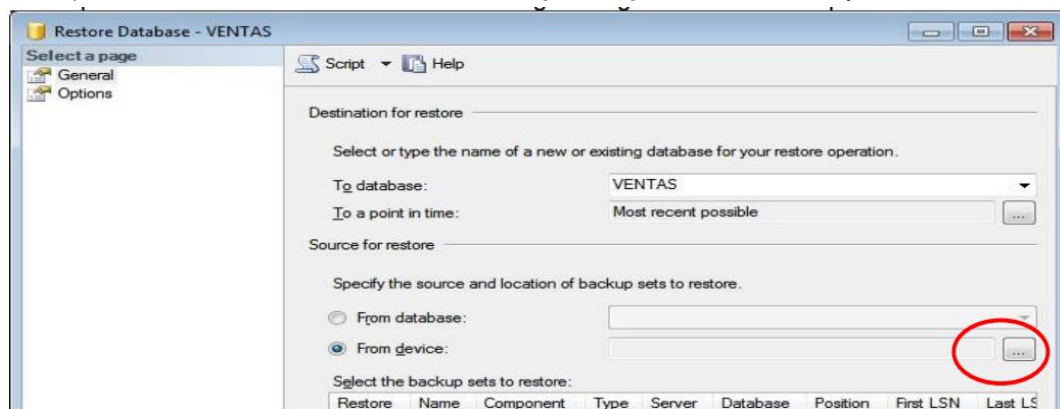


Figura N° 6 - 13: Restaurar Base de Datos seleccionando backup

6. Presionamos el botón OK.



Figura N° 6 - 14: Restaurar Base de Datos elegir base de datos

7. Seleccionamos **OK** nuevamente. Y en la Ventana **Restore Database – Ventas**, debemos marcar la opción **Restore** de nuestro archivo **backup** seleccionado. Luego hacemos clic sobre el botón **OK**.

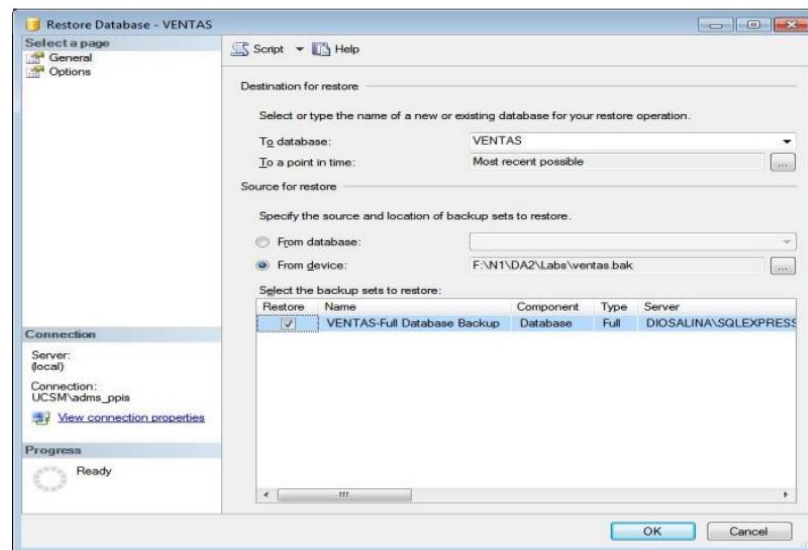


Figura N° 6 - 15: Restaurar Base de Datos finalización

8. La base de datos **VENTAS** estará restaurada.
- b) **CREANDO LA APLICACIÓN EN VISUAL STUDIO PARA EL REGISTRO DE CLIENTE**
1. Elija **Archivo/Nuevo/Sitio web** y dele el nombre de **MantenimientoBD** a su aplicación web.
 2. Diseñe el formulario como el que se muestra a continuación:

Figura N° 6 - 16: Interfaz de consulta

Control	Propiedad	Valor
TextBox1	(ID)	txtCodigo
TextBox2	(ID)	txtNombres
TextBox3	(ID)	txtDireccion
TextBox4	(ID)	txtTelefono
TextBox5	(ID)	txtMail
TextBox6	(ID)	txtEdad
Button1	(ID)	btnGrabar
	Text	Grabar
Label1	(ID)	lblResult

3. Haga doble clic sobre el botón **btnGrabar**, para modificar el código y añada el espacio de nombres.

```
using System.Data.SqlClient
```

4. Modifique el código como se muestra a continuación. Recuerde cambiar el **DataSource** por el nombre de su servidor:

```
public partial class _Default : System.Web.UI.Page
{
    string connectionString = "Data Source=localhost\\SQLEXPRESS;" +
        "Initial Catalog=VENTAS;Integrated Security=SSPI";
    protected void Page_Load(object sender, EventArgs e)
    {
        protected void btnGrabar_Click(object sender, EventArgs e)
        {
            string insertSQL = "INSERT INTO clientes VALUES(" +
                txtCodigo.Text + ", '" + txtNombres.Text + "', '" +
                txtDireccion.Text + "', '" + txtTelefono.Text + "', '" +
                txtMail.Text + "', " + txtEdad.Text + ")";
            SqlConnection con = new SqlConnection(connectionString);
            SqlCommand cmd = new SqlCommand(insertSQL, con);
            try
            {
                con.Open();
                cmd.ExecuteNonQuery();
                lblResult.Text = "Cliente registrado con éxito";
                con.Close();
                txtCodigo.Text = "";
                txtNombres.Text = "";
                txtDireccion.Text = "";
                txtTelefono.Text = "";
                txtMail.Text = "";
                txtEdad.Text = "";
            }
            catch (Exception err)
            {
                lblResult.Text = "Error al registrar el cliente";
                lblResult.Text += err.Message;
            }
        }
    }
}
```

5. Ejecute la aplicación y pruebe su funcionamiento ingresando datos para registrar a un cliente.

Figura N° 6 - 17: Consultar la Base de Datos

c) VALIDANDO LA INFORMACIÓN DE ENTRADA

- Usaremos los controles de validación proporcionados por **Visual Studio**. Para validar que el usuario siempre ingrese el código, nombre y dirección del cliente, al costado de cada control agregamos un control de validación **RequiredFieldValidator**. En cada control de validación agregado, debemos modificar la propiedad **ControlToValidate**, seleccionando el control correspondiente; y la propiedad **ErrorMessage** para indicar cuál será el mensaje mostrado, como se puede ver en la siguiente figura.

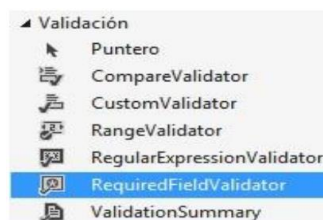


Figura N° 6 - 18: Elegir la validación

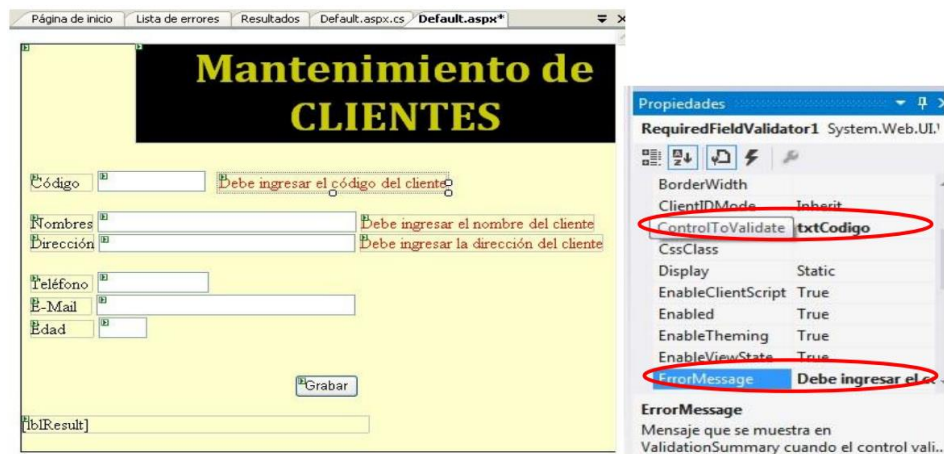


Figura N° 6 - 19: Restaurar Base de Datos

2. Añada el control de validación **RegularExpression** al costado del control **txtMail** y modifique las siguientes propiedades:



Propiedad		Valor
ControlToValidate		txtMail
ErrorMessage		El correo electrónico ingresado no es válido
ValidationExpression		\S+@\S+\.\S+

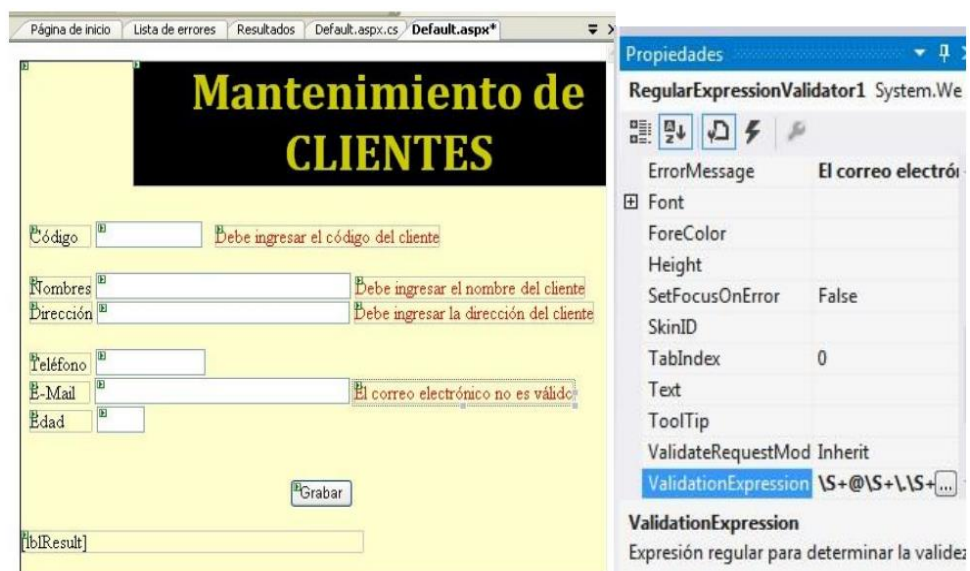


Figura N° 6 - 20: Configurar la validación

3. Añada el control de validación **RangeValidator** al costado del control **txtEdad** y modifique las siguientes propiedades:



Propiedad	Valor
ControlToValidate	txtEdad

ErrorMessage	La edad ingresada debe estar en el rango de 18 a 100.
MaximumValue	100
MinimumValue	18
Type	Integer



Figura N° 6 - 21: Restaurar Base de Datos

4. Ejecute la aplicación y observe lo siguiente:
 - Cuando visualiza el navegador, los mensajes de error no están visibles.
 - Si se ubica en el control **txtCodigo** y se mueve al siguiente control presionando la tecla TAB, sin ingresar un código para el cliente, el mensaje de error correspondiente será mostrado. Ocurre lo mismo si no cumplen las demás validaciones.
 - Si presiona el botón **Grabar** y no se cumplen alguna de las validaciones establecidas con los controles de validación, las acciones a ejecutarse serán ignoradas.
5. El problema surge si el navegador utilizado no soporta la validación en el lado del cliente (JavaScript). Las acciones del botón **Grabar** se ejecutarán a pesar de que la validación no sea exitosa. Para evitar esto, agregue la siguiente línea en el evento Clic del botón **Grabar**.

```

public partial class _Default : System.Web.UI.Page
{
    string connectionString = "Data Source=localhost\\SQLEXPRESS;" +
        "Initial Catalog=VENTAS;Integrated Security=SSPI";

    protected void Page_Load(object sender, EventArgs e) {...}
    protected void btnGrabar_Click(object sender, EventArgs e)
    {
        //Aborta si la página no es valida
        if (!this.IsValid) return;
        string insertSQL = "INSERT INTO clientes VALUES(" +
            txtCodigo.Text + ", '" + txtNombres.Text + "', '" +
            txtDireccion.Text + "', '" + txtTelefono.Text + "', '" +
            txtMail.Text + "', " + txtEdad.Text + ")";
        SqlConnection con = new SqlConnection(connectionString);
        SqlCommand cmd = new SqlCommand(insertSQL, con);
    }
}

```

6. Ahora añada el control **ValidationSummary** al final del formulario, y modifique la propiedad **Display** de todos los controles de validación a **None**.



7. Ejecuta la aplicación y verifique que los mensajes de error de todos los controles de validación, esta vez son mostrados en el control **ValidationSummary**.

Mantenimiento de CLIENTES

Código

Nombres

Dirección

Teléfono

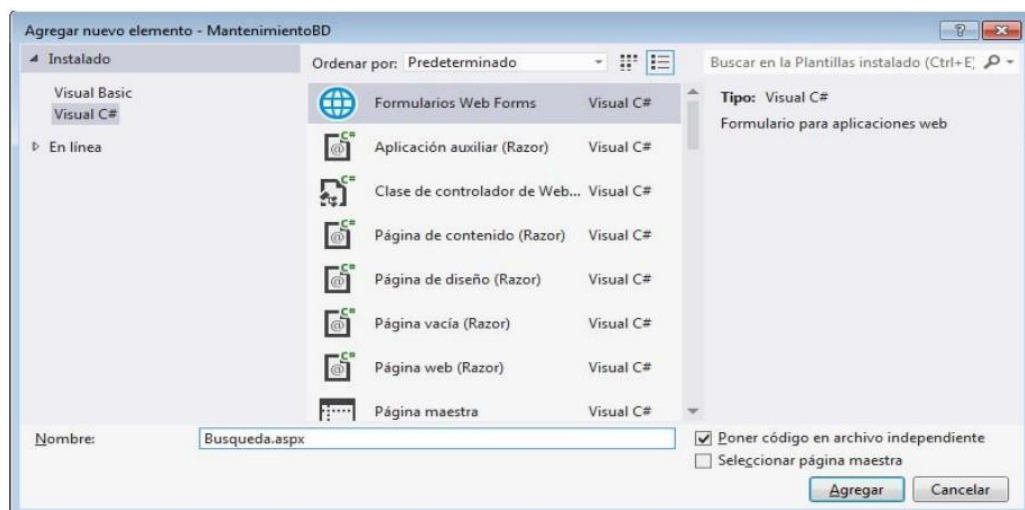
E-Mail

Edad

- Debe ingresar el código del cliente
- Debe ingresar el nombre del cliente
- Debe ingresar la dirección del cliente
- El correo electrónico no es válido
- La edad ingresada debe estar en el rango de 18 a 100.

d) BÚSQUEDA DE DATOS

1. Añada un formulario con el nombre Busqueda.aspx



2. Modifique el diseño del formulario para que luzca como se muestra a continuación:

Búsqueda de CLIENTES

Código

Nombres

Dirección

Teléfono

E-Mail

Edad

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

[blResult]

Control	Propiedad	Valor
TextBox1	(ID)	txtCodigo
	ReadOnl	yFalse
TextBox2	(ID)	txtNombres
	ReadOnly	True
TextBox3	(ID)	txtDireccion
	ReadOnly	True
TextBox4	(ID)	txtTelefono
	ReadOnly	True
TextBox5	(ID)	txtMail
	ReadOnly	True
TextBox6	(ID)	txtEdad
	ReadOnly	True
Button1	(ID)	btnBuscar
	Text	Buscar
Button2	(ID)	btnLimpiar
	Text	Limpiar
Button3	ID)	btnMostrar
	Text	Mostrar Todos los Clientes
Label1	(ID)	lblResult
GridView	(ID)	GridView1

3. En el código, añada el espacio de nombres:

```
using System.Data.SqlClient;
```

4. Haga doble clic sobre el botón Buscar y añada el siguiente código:

```
protected void btnBuscar_Click(object sender, EventArgs e)
{
    lblResult.Text = "";
    string selectSQL = "SELECT * FROM clientes where codigo=" + txtCodigo.Text;
    SqlConnection con = new SqlConnection(connectionString);
    SqlCommand cmd = new SqlCommand(selectSQL, con);
    SqlDataReader dr;
    try
    {
        con.Open();
        dr = cmd.ExecuteReader();
        if (dr.Read())
        {
            txtNombres.Text = dr[1].ToString();
            txtDireccion.Text = dr["direccion"].ToString();
            txtTelefono.Text = dr[3].ToString();
            txtMail.Text = dr[4].ToString();
            txtEdad.Text = dr[5].ToString();
        }
        con.Close();
        dr.Close();
    }
    catch (Exception err)
    {
        lblResult.Text = "Error al registrar el cliente";
        lblResult.Text += err.Message;
    }
}
```

5. Haga doble clic sobre el botón Limpiar y añada el siguiente código:

```
protected void btnLimpiar_Click(object sender, EventArgs e)
{
    txtCodigo.Text = "";
    txtNombres.Text = "";
    txtDireccion.Text = "";
    txtTelefono.Text = "";
    txtMail.Text = "";
    txtEdad.Text = "";
}
```

6. Haga doble clic sobre el botón **Mostrar Todos los Clientes** y añada el siguiente código:

```
protected void btnMostrar_Click(object sender, EventArgs e)
{
    string selectSQL = "SELECT * FROM clientes";
    SqlConnection con = new SqlConnection(connectionString);
    SqlCommand cmd = new SqlCommand(selectSQL, con);
    SqlDataReader dr;
    try
    {
        con.Open();
        dr = cmd.ExecuteReader();
        GridView1.DataSource = dr;
        GridView1.DataBind();
        con.Close();
    }
    catch (Exception err)
    {
        lblResult.Text = "Error al registrar el cliente";
        lblResult.Text += err.Message;
    }
}
```

7. Ejecute la aplicación y verifique su funcionamiento. Ingrese un código a buscar (Ejemplo: 1) y presione el botón Buscar para ver la información recuperada desde la Base de Datos. Presione el botón Mostrar Todos los Clientes, para ver un listado completo de todos los Clientes registrados.

The screenshot shows a web browser window with the address `http://localhost:2839/BaseDatos/Busqueda.aspx`. The page has a green background and a black header with the text 'Búsqueda de CLIENTES' in yellow. Below the header is a search form with the following fields:

- Código:** A text box containing '1', with 'Buscar' and 'Limpiar' buttons next to it.
- Nombres:** A text box containing 'Juan Perez'.
- Dirección:** A text box containing 'Los Portales 310'.
- Teléfono:** A text box containing '959663322'.
- E-Mail:** A text box containing 'jperez@hotmail.com'.
- Edad:** A text box containing '30'.

Below the form is a button labeled 'Mostrar Todos los Clientes'. Below the button is a table with the following data:

codigo	nombre	direccion	telefono	email	edad
1	Juan Perez	Los Portales 310	959663322	jperez@hotmail.com	30
2	José Diaz	Los Angeles de Cayma 250	959808080	jdiaz@hotmail.com	35
3	Roberto Gonzales	Las Orquideas 500	959757878	rgonzales@hotmail.com	38
4	Ruben Peralta	Los Portales 510	959663322	rperalta@hotmail.com	30
5	Maria de los Angeles	Juan Pablo V. y Guzman L. 5	959272754	angeles@hotmail.com	30
6	Arturo Ordoñez	Los Jasmínes 250	959707074	aordonez@hotmail.com	32
7	Josefa Lopez	Los Portales 10	959202023	jlopez@hotmail.com	25
8	Sandra Ramirez	Palacio Viejo 100	958757575	sramirez@hotmail.com	28
9	Pedro Lopez	Consuelo 440	959695656	plopez@hotmail.com	30

Figura N° 6 - 22: Resultados e consulta a la Base de Datos

V

EJERCICIOS PROPUESTOS

1. Cree la tabla PRODUCTOS en la base de datos VENTAS con los siguientes campos: Tabla PRODUCTOS

Nombre de columna (Column Name)	Tipo (Data Type)
codpro	int
nompro	nvarchar(50)
precio	decimal(6,2)
stock	decimal(6,2)

2. Inserte por lo menos 10 registros en la tabla creada.

Ejemplo:

```
INSERT INTO PRODUCTOS VALUES (1, 'Impresora HP', 100, 10);
```

3. Implemente una aplicación que permita mostrar el listado de los productos registrados en la base de datos VENTAS.
4. Diseñe una base de datos para un sistema académico considerando Alumno, Cursos, Horario y Docente e impleméntelo en el SQL Server, agregando registros de datos a las tablas.

V

CUESTIONARIO

1. ¿Qué es un dato?
2. ¿Cómo se organizan los datos para su almacenamiento?
3. ¿Qué operaciones se realizan sobre los datos para su gestión?
4. ¿Qué es un campo?
5. ¿Qué es un registro?
6. ¿Qué es una Tabla?
7. ¿Qué es una tabla indexada?
8. ¿Qué es un índice?
9. ¿Qué es una clave principal?
10. ¿Qué es una clave foránea?
11. ¿Qué es una Base de datos?
12. ¿Qué es un gestor de Bases de Datos?
13. ¿Cómo se diseña el almacenamiento de información?
14. ¿Cuál es el objetivo de la utilización de las formas normales?
15. ¿Cómo se aplican las formas normales para el diseño de bases de datos?
16. ¿Qué es ODBC?
17. ¿Qué es DAO?
18. ¿Qué es el ADO.NET?
19. ¿Qué es SQL?
20. ¿Cuáles son los principales comandos SQL?
21. ¿Qué es una cadena de conexión?
22. ¿Qué es un DataSet?
23. ¿Cómo funciona un dataset?
24. ¿Qué controles se usan para interactuar con un dataset?
25. ¿Qué es el álgebra relacional?

VI

BIBLIOGRAFIA Y REFERENCIAS

- Microsoft Official Course, Developing Web Applications using Microsoft Visual Studio 2010, Microsoft Corporation, 2010
- Sánchez Flores, "Desarrollado aplicaciones con Visual C# NET 2008". Ed. Macro, Perú, 2008
- Joseph Albarari y Ben Albarari, C# 4.0 in a Nutshell, O'Reilly Media., 2010
- Matthew MacDonald, "Beginning ASP.NET 2.0 in C# 2005", Ed. Apress, USA, 2006.