Guía para

entender SQL Server Y consejos

Índice

[Crear base de datos 2](#_Toc177611229)

[Procedimientos almacenados para cada caso 3](#_Toc177611230)

[Vistas 4](#_Toc177611231)

[Estructura 5](#_Toc177611232)

[Conexión 5](#_Toc177611233)

[FORM 6](#_Toc177611234)

[Form objetivo 7](#_Toc177611235)

[CRUD 7](#_Toc177611236)

[DATAGRID 11](#_Toc177611237)

[Colores 13](#_Toc177611238)

[JOB 14](#_Toc177611239)

### Crear base de datos

-- Crear la base de datos

CREATE DATABASE Visual\_Basic\_CRUD;

GO

-- Usar la base de datos

USE Visual\_Basic\_CRUD;

GO

-- Crear la tabla Estado

CREATE TABLE Estado (

EstadoID INT PRIMARY KEY IDENTITY(1,1),

Estado NVARCHAR(50) NOT NULL

);

GO

-- Crear la tabla Tienda

CREATE TABLE Tienda (

TiendaID INT PRIMARY KEY IDENTITY(1,1),

Nombre NVARCHAR(100) NOT NULL,

Ubicacion NVARCHAR(100) NOT NULL,

EstadoID INT NOT NULL,

FOREIGN KEY (EstadoID) REFERENCES Estado(EstadoID)

);

GO

-- Crear la tabla Empleado

CREATE TABLE Empleado (

EmpleadoID INT PRIMARY KEY IDENTITY(1,1),

Nombre NVARCHAR(100) NOT NULL,

TiendaID INT NOT NULL,

EstadoID INT NOT NULL,

FOREIGN KEY (TiendaID) REFERENCES Tienda(TiendaID),

FOREIGN KEY (EstadoID) REFERENCES Estado(EstadoID)

);

GO

-- Crear la tabla Producto

CREATE TABLE Producto (

ProductoID INT PRIMARY KEY IDENTITY(1,1),

Nombre NVARCHAR(100) NOT NULL,

EstadoID INT NOT NULL,

FOREIGN KEY (EstadoID) REFERENCES Estado(EstadoID)

);

GO

--Inserts para estado

INSERT INTO Estado(Estado) VALUES (1);

INSERT INTO Estado(Estado) VALUES (2);

### Procedimientos almacenados para cada caso

CREATE PROCEDURE SP\_AGREGAR\_EMPLEADO

@NOMBRE NVARCHAR(100),

@TIENDA INT

AS

BEGIN

INSERT INTO Empleado (Nombre,TiendaID,EstadoID)

VALUES (@NOMBRE, @TIENDA,1);

END;

GO

CREATE PROCEDURE SP\_BUSCAR\_EMPLEADO

@Buscar NVARCHAR(100)

AS

BEGIN

SELECT

Empleado.EmpleadoID AS [#],

Empleado.Nombre AS [Nombre],

Empleado.TiendaID as [ID\_de\_la\_tienda],

Tienda.Nombre AS [Nombre\_de\_la\_Tienda]

FROM

Empleado INNER JOIN

Tienda ON Empleado.TiendaID = Tienda.TiendaID WHERE Empleado.EstadoID = 1 AND (CAST (Empleado.EmpleadoID AS nvarchar) + Empleado.Nombre + Tienda.Nombre) Like '%' + @Buscar + '%';

END;

GO

CREATE PROCEDURE SP\_MODIFICAR\_EMPLEADO

@EMPLEADOID INT,

@NOMBRE NVARCHAR(100),

@TIENDAID INT

AS

BEGIN

UPDATE Empleado SET

EMPLEADO.Nombre = @NOMBRE,

EMPLEADO.TiendaID = @TIENDAID

WHERE EMPLEADO.EmpleadoID = @EMPLEADOID

END;

GO

CREATE PROCEDURE SP\_ELIMINAR\_EMPLEADO

@EMPLEADOID INT

AS

BEGIN

UPDATE Empleado SET

EMPLEADO.EstadoID = 2

WHERE EMPLEADO.EmpleadoID = @EMPLEADOID

END;

GO

### Vistas

CREATE VIEW V\_MOSTRAR\_EMPLEADO AS

SELECT

Empleado.EmpleadoID AS [#],

Empleado.Nombre AS [Nombre],

Empleado.TiendaID as [ID de la tienda],

Tienda.Nombre AS [Nombre Tienda]

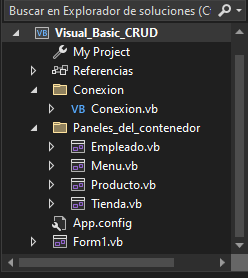
FROM

Empleado INNER JOIN

Tienda ON Empleado.TiendaID = Tienda.TiendaID WHERE Empleado.EstadoID = 1;

GO

### Estructura



### Conexión

'LIBRERIAS NECESARIAS

Imports System.Data

Imports System.Data.SqlClient

Imports System.Configuration

Module Conexion

'CADENA DE CONEXION'

Public conexionsql As New SqlConnection("Server=DESKTOP-V683HQU\SQLEXPRESS; Database=Visual\_Basic\_CRUD; User Id=sqlserver; Password=root;")

'ABRIR Y CERRAR CONEXIONES'

Sub Abrir\_Conexion()

If conexionsql.State = 0 Then

conexionsql.Open()

End If

End Sub

Sub Cerrar\_Conexion()

If conexionsql.State = 1 Then

conexionsql.Close()

End If

End Sub

End Module

### FORM

Public Sub CargarFormularioEnPanel(formulario As Form)

' Limpia el panel antes de agregar un nuevo formulario

Contenedor.Controls.Clear()

' Establece el tamaño del formulario para que se ajuste al panel

formulario.TopLevel = False

formulario.FormBorderStyle = FormBorderStyle.None

formulario.Dock = DockStyle.Fill

' Agrega el formulario al panel

Contenedor.Controls.Add(formulario)

formulario.Show()

End Sub

Private Sub Button1\_MouseClick(sender As Object, e As MouseEventArgs) Handles Button1.MouseClick

Dim frmEmpleado As New Empleado()

' Llamar a la función para cargar el formulario en el panel

CargarFormularioEnPanel(frmEmpleado)

End Sub

### Form objetivo

1. Libreria

Imports System.Data.SqlClient

2. Limpiar txt

Private Sub Limpiar()

txtNombreEmpleado.Clear()

txtIDUbucacion.Clear()

End Sub

3. Funcionalidad de botón agregar y cerrar

Private Sub Button1\_MouseClick(sender As Object, e As MouseEventArgs) Handles Button1.MouseClick

PanelAgregar.Visible = True

Limpiar()

btnAgregar.Enabled = True

btnModificar.Enabled = False

txtNombreEmpleado.Focus()

End Sub

Private Sub Button4\_MouseClick(sender As Object, e As MouseEventArgs) Handles Button4.MouseClick

Limpiar()

PanelAgregar.Visible = False

btnAgregar.Enabled = True

btnModificar.Enabled = True

End Sub

### CRUD

1. AGREGAR

Private Sub btnAgregar\_MouseClick(sender As Object, e As MouseEventArgs) Handles btnAgregar.MouseClick

Dim Consulta As New SqlCommand

If txtNombreEmpleado.Text <> "" And txtIDUbucacion.Text <> "" Then

Try

Abrir\_Conexion()

Consulta = New SqlCommand("SP\_AGREGAR\_EMPLEADO", conexionsql)

Consulta.CommandType = 4

Consulta.Parameters.AddWithValue("@Nombre", txtNombreEmpleado.Text.ToString)

Consulta.Parameters.AddWithValue("@TIENDA", Convert.ToInt32(txtIDUbucacion.Text))

Consulta.ExecuteNonQuery()

Cerrar\_Conexion()

PanelAgregar.Visible = False

Limpiar()

Mostrar()

Catch ex As Exception

End Try

Else

MsgBox("Los capos son obligatorios")

End If

End Sub

2. PROGRAMACION PARA MOSTRAR

Sub Mostrar()

Dim dt As New DataTable

Dim da As SqlDataAdapter

Try

Abrir\_Conexion()

'Forma para traer la data desde una vista'

da = New SqlDataAdapter("SELECT \* FROM V\_MOSTRAR\_EMPLEADO", conexionsql)

'Forma para traer la data desde un procedimiento'

'da = New SqlDataAdapter("SP\_Mostrar\_Tienda", conexionsql)

da.Fill(dt)

DataGridEmpleado.DataSource = dt

Cerrar\_Conexion()

'Cambiar ancho del datagrid'

DataGridEmpleado.Columns(0).Width = 70

DataGridEmpleado.Columns(1).Width = 20

DataGridEmpleado.Columns(2).Width = 200

DataGridEmpleado.Columns(3).Width = 200

DataGridEmpleado.Columns(4).Width = 200

'Apariencia de los encabezados del datagrid'

DataGridEmpleado.EnableHeadersVisualStyles = False

Dim estilo As DataGridViewCellStyle = New DataGridViewCellStyle()

estilo.BackColor = Color.White

estilo.ForeColor = Color.Black

estilo.Font = New Font("Arial", 10, FontStyle.Regular Or FontStyle.Bold)

DataGridEmpleado.ColumnHeadersDefaultCellStyle = estilo

Catch ex As Exception

End Try

End Sub

3. MOSTRAR EN FORM AL CARGAR LA PESTAÑA

Private Sub Empleado\_Load(sender As Object, e As EventArgs) Handles Me.Load

Mostrar()

End Sub

4. PROGRAMACION PARA BUSCAR EN TIEMPO REAL

Private Sub BuscarDG()

Dim dt As New DataTable

Dim da As SqlDataAdapter

Try

Abrir\_Conexion()

'Forma para traer la data desde una vista'

da = New SqlDataAdapter("SP\_BUSCAR\_EMPLEADO", conexionsql)

da.SelectCommand.CommandType = 4

da.SelectCommand.Parameters.AddWithValue("@Buscar", txtBuscar.Text)

da.Fill(dt)

DataGridEmpleado.DataSource = dt

Cerrar\_Conexion()

'Cambiar ancho del datagrid'

DataGridEmpleado.Columns(0).Width = 70

DataGridEmpleado.Columns(1).Width = 20

DataGridEmpleado.Columns(2).Width = 200

DataGridEmpleado.Columns(3).Width = 200

DataGridEmpleado.Columns(4).Width = 200

'Apariencia de los encabezados del datagrid'

DataGridEmpleado.EnableHeadersVisualStyles = False

Dim estilo As DataGridViewCellStyle = New DataGridViewCellStyle()

estilo.BackColor = Color.White

estilo.ForeColor = Color.Black

estilo.Font = New Font("Arial", 10, FontStyle.Regular Or FontStyle.Bold)

DataGridEmpleado.ColumnHeadersDefaultCellStyle = estilo

Catch ex As Exception

End Try

End Sub

5. ACTIVAR FUNCIONALIDAD DE BUSCAR EN TIEMPO REAL

Private Sub txtBuscar\_TextChanged(sender As Object, e As EventArgs) Handles txtBuscar.TextChanged

BuscarDG()

End Sub

6. AL DARLE DOBLE CLICK QUE META LOS DATOS EN LOS TXT

Private Sub DataGridEmpleado\_DoubleClick(sender As Object, e As EventArgs) Handles DataGridEmpleado.DoubleClick

PanelAgregar.Visible = True

btnAgregar.Enabled = False

btnModificar.Enabled = True

Try

Dim ID As String

TextID.Text = DataGridEmpleado.SelectedCells.Item(1).Value

txtNombreEmpleado.Text = DataGridEmpleado.SelectedCells.Item(2).Value

txtIDUbucacion.Text = DataGridEmpleado.SelectedCells.Item(3).Value

Catch ex As Exception

End Try

End Sub

7. MODIFICAR

Private Sub btnModificar\_MouseClick(sender As Object, e As MouseEventArgs) Handles btnModificar.MouseClick

Dim Consulta As New SqlCommand

If txtNombreEmpleado.Text <> "" And txtIDUbucacion.Text <> "" Then

Try

Abrir\_Conexion()

Consulta = New SqlCommand("SP\_MODIFICAR\_EMPLEADO", conexionsql)

Consulta.CommandType = 4

Consulta.Parameters.AddWithValue("@EMPLEADOID", TextID.Text.ToString)

Consulta.Parameters.AddWithValue("@NOMBRE", txtNombreEmpleado.Text.ToString)

Consulta.Parameters.AddWithValue("@TIENDAID", txtIDUbucacion.Text.ToString)

Consulta.ExecuteNonQuery()

Cerrar\_Conexion()

PanelAgregar.Visible = False

Limpiar()

Mostrar()

Catch ex As Exception

End Try

Else

MsgBox("Los capos son obligatorios")

End If

End Sub

8. ELIMINAR

Private Sub DataGridEmpleado\_CellContentClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridEmpleado.CellContentClick

'Verificar si se ha dado click sobre la columna eliminar'

If e.ColumnIndex = DataGridEmpleado.Columns.Item("Eliminar").Index Then

Dim result As DialogResult

result = MsgBox("El registro sera eliminado", vbQuestion + vbOKCancel, "Empleados")

If result = DialogResult.OK Then

Dim Consulta As SqlCommand

Try

Abrir\_Conexion()

Consulta = New SqlCommand("SP\_ELIMINAR\_EMPLEADO", conexionsql)

Consulta.CommandType = 4

Consulta.Parameters.AddWithValue("@EMPLEADOID", DataGridEmpleado.SelectedCells.Item(1).Value)

Consulta.ExecuteNonQuery()

Cerrar\_Conexion()

Mostrar()

Catch ex As Exception

End Try

Else

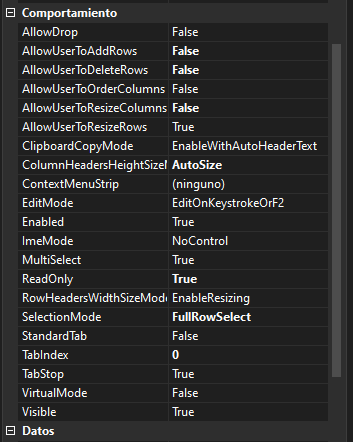
MsgBox("Eliminacion Cancelada")

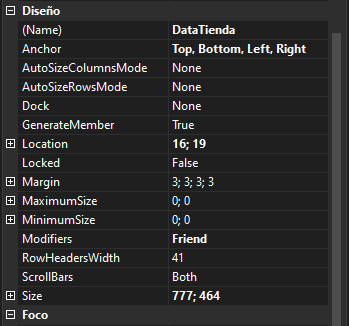
End If

End If

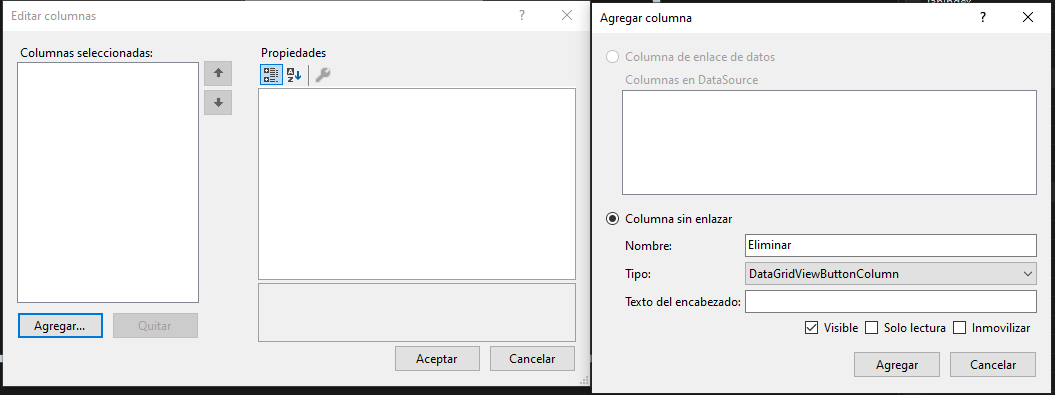
End Sub

### DATAGRID









### Colores

Gris claro

31; 31; 31



Gris oscuro

26; 26; 26



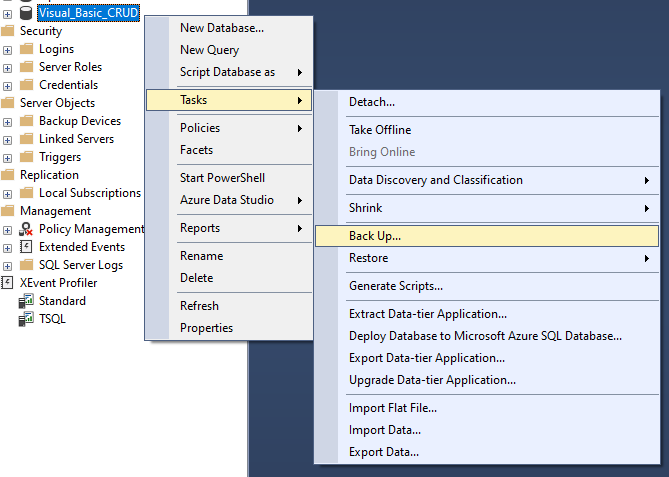
Mouse encima del botón

37; 37; 37

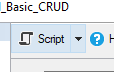


### JOB

1. Obtener el SCRIPT

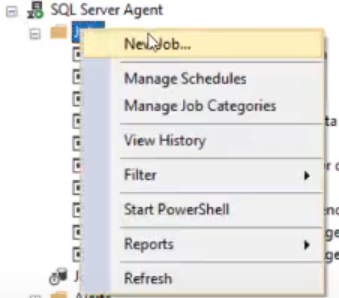


2. Darle a script para que me genere el script

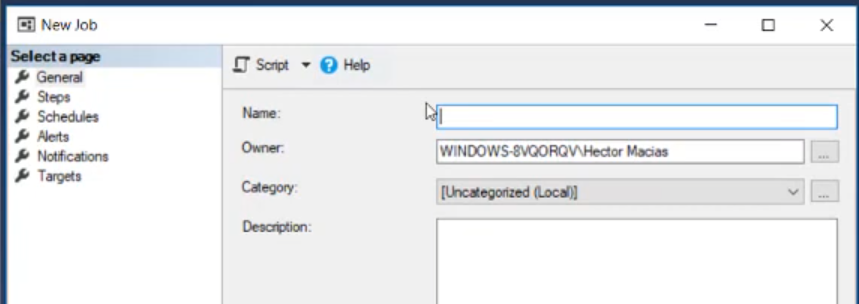


3. Compiar el script

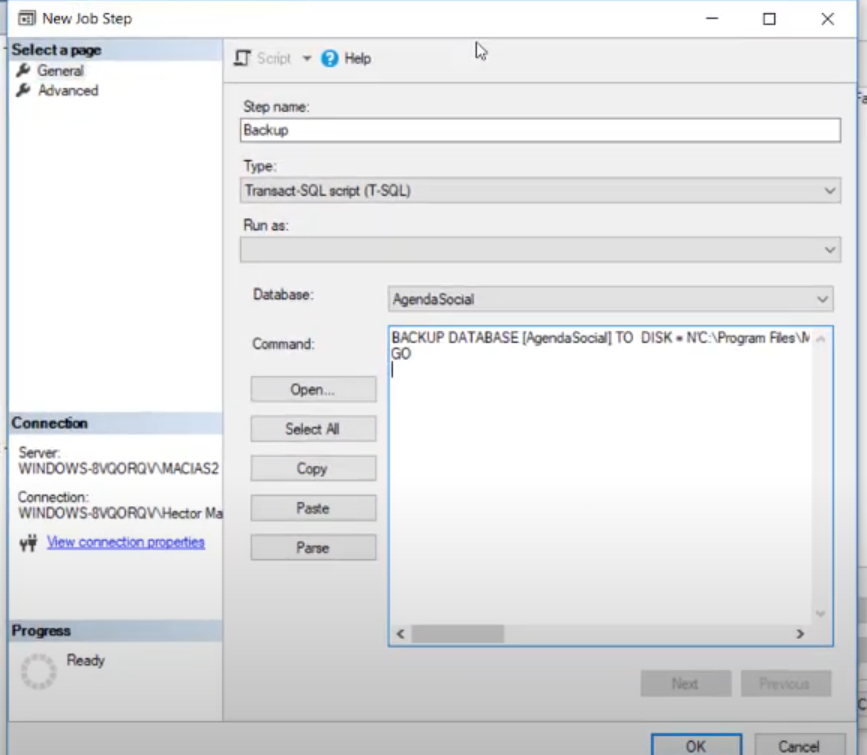
4. Crear el job



5. Ponerle nombre al JOB

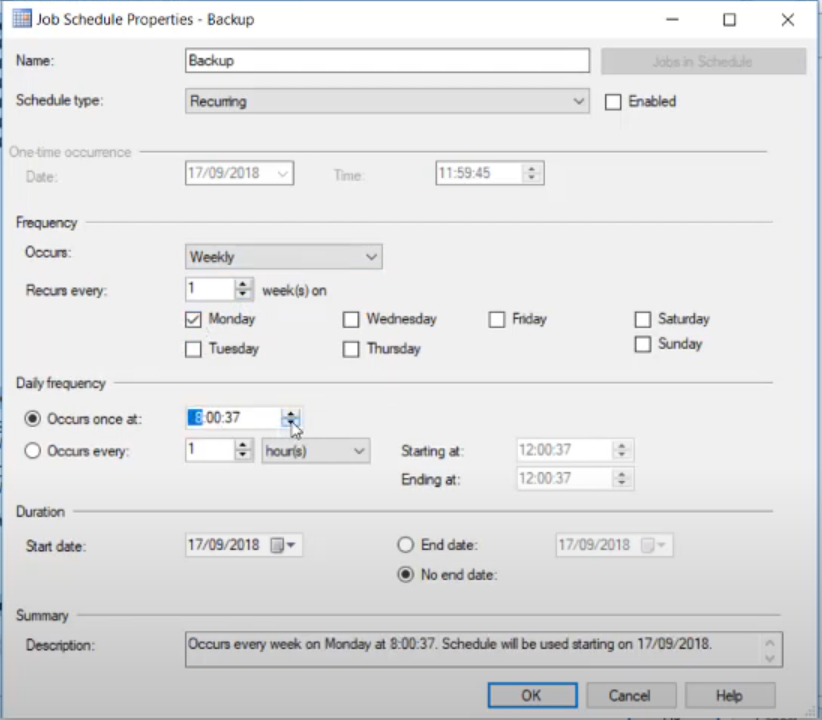


6. STEPS -> NEW



* Step name
* Database
* Codigo que genere anteriormente
  + Parse para probarlo

7. Schedule



Programar recurrencia, días, hora,