# Como agregar un SP Custom que retorna una entidad del modelo

El tema es que se ha decidido hacer un SP custom que retorna TODOS los ATRIBUTOS de una entidad entendiendo para el caso de negocio planteado que las consultas estándar del framework (FindBY, FindById, FindAll y FinbByRange) no resuelven el problema planteado

Una vez que tenemos el SP, en nuestro ejemplo va a ser **SP\_FindByCustom**, en el proyecto **NombreProyecto.Data** agregamos la clase NombreClaseEntity.cs, que es una clase parcial de NombreClaseEntity.cs que se encuentra en la carpeta Generados.

Dentro de esta clase generamos el método que va a hacer la consulta.

Ejemplo:

using System;

using System.Collections.Generic;

using System.Text;

using Circo.Framework.Persistencia;

using Circo.Framework.Persistencia.Filters;

namespace NombreProyecto.Data.Entities

{

public partial class NombreClaseEntity

{

public static List<NombreClaseEntity> FindByCustom(List<EntityFilter> filters)

{

return CommonFindBy("SP\_FindByCustom", filters, null);

}

}

}

En el proyecto NombreProyecto.BusinessLogic en la carpeta Handlers agregamos la clase NombreClaseHandler.cs.

En esta clase creamos el método que retornara la lista de NombreClaseView.

**Ejemplo:**

using System;

using System.Collections.Generic;

using System.Text;

using System.Data;

using Circo.Framework.BusinessLogic;

using Circo.Framework.BusinessLogic.Views.Filters;

using Circo.Framework.BusinessLogic.Handlers;

using NombreProyecto.Data.Entities;

using NombreProyecto.BusinessLogic.Views;

namespace NombreProyecto.BusinessLogic.Handlers

{

public partial class NombreClaseHandler

{

public static List<NombreClaseView> FindByCustom (List<ViewFilter> filters)

{

List<NombreClaseView> vs = new List<NombreClaseView>();

List<NombreClaseEntity> entities = CustomerEntity.FindByUser(Filter.MapViewFiltersToEntityFilters(filters));

foreach (NombreClaseEntity en in entities )

vs.Add(new NombreClaseView(en));

return vs;

}

}

}

**Para hacer la consulta:**

List<ViewFilter> filter = new List<ViewFilter>();

filter.Add(new ViewCustomFilter("IdUser", DbType.Int32, 1));

List<NombreClaseView> customs = NombreClaseHandler.FindByCustom(filter);

# Cómo guardar una imagen en la BD con Circo

Generador de código.

Para realizar la demostración utilizaremos la entidad **LogoProveedor** a modo de ejemplo. Esta entidad contendrá los siguientes campos (Ver Imagen 1).

* **Imagen:** Campo del tipo [binary] en el que se almacenará la imagen.
* **Nombre:** Nombre de la imagen en el cual se va a incluir la extensión (ejemplo “Foto.jpg”)

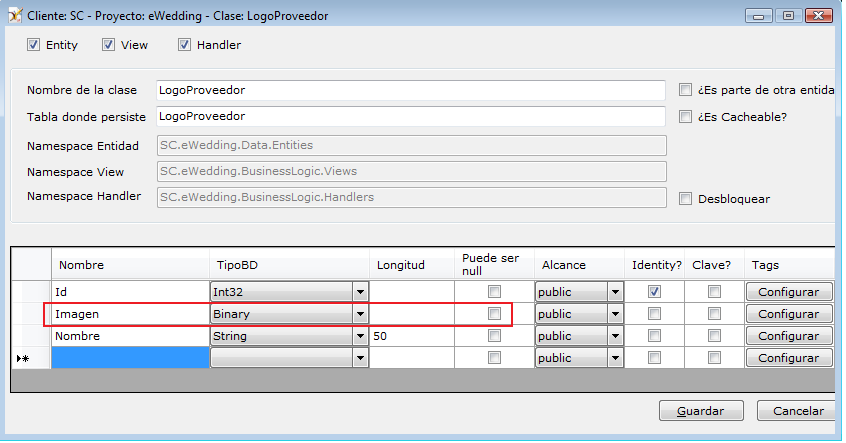


Imagen 1

Como guardar la imagen en la BD:

public void GuardarImagen()

{

string pathImg = @"C:\Temp\Foto.jpg";

LogoProveedorView lp = new LogoProveedorView();

lp.Nombre = "Foto.jpg";

lp.Imagen = GetByteFile(pathImg);

LogoProveedorHandler.Persist(lp);

}

Para guardar la imagen solo necesitamos convertirla al tipo binary para ello utilizamos el método **GetByteFile** y el nombre de la imagen para volver a abrirla.

static byte[] GetByteFile(string filename)

{

try

{

FileStream fs = new FileStream(filename, FileMode.Open, FileAccess.Read);

byte[] buffer = new byte[(int)fs.Length];

fs.Read(buffer, 0, buffer.Length);

fs.Close();

return buffer;

}

catch (Exception ex)

{

throw new Exception("LoadFile - fileName: " + filename + " - " +

ex.Message);

}

}

**GetByteFile:** Este método a partir de la ruta de imagen (filename) devuelve un buffer del tipo Byte[] que es el que utilizaremos para almacenar la imagen en la Base.

Como obtenemos la imagen de la BD.

private System.Drawing.Image GetImageFromModel(int id)

{

System.Drawing.Image image;

try

{

LogoProveedorView foto = LogoProveedorHandler.FindById(id);

System.IO.MemoryStream ms = new System.IO.MemoryStream(foto.Imagen);

image = System.Drawing.Image.FromStream(ms);

}

catch (Exception ex)

{

throw new Exception("GetImageFromModel - Id: " + id.ToString() + " - " +

ex.Message);

}

return image;

}

Este método a partir del Id de LogoProveedor arma la imagen que es la que mostraremos en la pantalla.

## Tabla

## 

## Stored Procedures

Insertar Nuevo

CREATE PROCEDURE [dbo].[f\_LogoProveedor\_New]

(

@pImagen VARBINARY(MAX),

@pNombre VARCHAR(50),

@pId INT OUTPUT

)

AS

SET NOCOUNT ON

INSERT INTO

[LogoProveedor]

(

[LogoProveedor].[Imagen],

[LogoProveedor].[Nombre]

)

VALUES

(

@pImagen,

@pNombre

)

Actualizar

CREATE PROCEDURE [dbo].[f\_LogoProveedor\_Update]

(

@pId INT,

@pImagen VARBINARY(MAX),

@pNombre VARCHAR(50)

)

AS

SET NOCOUNT ON

UPDATE

[LogoProveedor]

SET

[LogoProveedor].[Imagen] = @pImagen,

[LogoProveedor].[Nombre] = @pNombre

WHERE

[LogoProveedor].[Id] = @pId

# Como retornar datos de un SP que no devuelve una entidad

public static DataTable FindBySearchParametersNoFW(int idProject, int idProduct, int idProductAll, string keyword)

{

DbCommand dbCommand = Conn.CurrentDB.GetStoredProcCommand("pkg\_Customs.c\_SearchIconRequests");

if (idProject != 0)

Conn.CurrentDB.AddInParameter(dbCommand, "pIdProject", DbType.Int32, idProject);

if (idProduct != 0)

Conn.CurrentDB.AddInParameter(dbCommand, "pIdProduct", DbType.Int32, idProduct);

if (idProduct != 0)

Conn.CurrentDB.AddInParameter(dbCommand, "pIdProductAll", DbType.Int32, idProductAll);

//Conn.CurrentDB.AddInParameter(dbCommand, "pIdLocation", DbType.Int32, DBNull.Value );

if (keyword != "")

Conn.CurrentDB.AddInParameter(dbCommand, "pKeyword", DbType.String, keyword);

return Conn.CurrentDB.ExecuteDataSet(dbCommand).Tables[0];

}