**Como llamar a los stored procedure de la base de datos.**

1. **Pasándole el nombre del procedure mas las variables que necesita.**

Primero le seteo una lista con los parametros que va a necesitar el SP. Por ejemplo el username. La variabla @Username tiene que tener el mismo nombre que cuando la defini en el SP. Este this.Username puede ser el username que ingresaron en el textbox de la app. This.Username = txbUsername.Text()

parameterList.Add(new SqlParameter("@Username", this.Username)

Luego ejecuto esta linea, donde al ExecuteDataSet le paso el nombre del procedure, el tipo de cosa que le estoy pasando, y la lista de parametros.

DataSet ds = SQLHelper.ExecuteDataSet("traerUsuarioActivoPorUsername", CommandType.StoredProcedure, parameterList);

Luego se ejecuta este store procedure, el cual va a devolver una consulta, la tabla con el usuario que tenga el username que le pase por parametro.

ALTER PROCEDURE [OOZMA\_KAPPA].[traerUsuarioActivoPorUsername]

@Username nvarchar(255)

AS

SELECT \*

FROM OOZMA\_KAPPA.Usuario

WHERE usuario\_username = @Username AND usuario\_estado = 0;

Es por eso que defino un DataSet ds para guardar el resultado del sp.

Una vez que tengo guardada la “tabla” en el ds, tengo que desarmarla y guardarla en variables de C# para poder usarlos. No puedo usar directamente el ds , si no que tengo que asignarle cada columna a una variable.

En el caso de la clase Usuario al principio defini un usuario del tipo Usuario. Por lo que los datos obtenidos por el sp los voy a guardar en este objeto usuario. Para ello utilizo y defino los metodos DATAROWTOOBJECT (convertir fila en objeto basicamente).

public override void DataRowToObject(DataRow dr)

{

// Esto es tal cual lo devuelve el stored de la DB

this.usuario\_id = Convert.ToInt32(dr["usuario\_id"]);

this.Username = dr["usuario\_username"].ToString();

this.Password = dr["usuario\_password"].ToString();

this.Estado = Convert.ToBoolean(dr["usuario\_estado"]);

this.ClaveAutoGenerada = esClaveAutoGenerada(dr);

}

Y asi obtuve los datos que queria de la bd. Ya los puedo usar para lo que quiera.

1. **Funciones que arman store procedure**

En la clase Base tengo definidos los siguientes encabezados para los SP

protected string \_strInsertar = "insert";

protected string \_strModificar = "update";

protected string \_strEliminar = "delete";

protected string \_strDeshabilitar = "deshabilitar";

protected string \_strTraerListado = "traerListado";

protected string \_strRetornoID = "\_RetornarID";

Usando esta forma directamente le paso a la entidad en la que estoy, la accion que deseo realizar en la base y los parametros que va a necesitar mi sp. Por ejemplo: si quiero que me traiga los roles que tiene un usuario hago.

parameterList.Add(new SqlParameter("@Username", this.Username)

DataSet ds = unRol.TraerListado(unRol.parameterList, "PorId\_Usuario");

Defino un DataSet ds para guardar el resultado del sp.

Una vez que tengo guardada la “tabla” en el ds, tengo que desarmarla y guardarla en variables de C# para poder usarlos. No puedo usar directamente el ds , si no que tengo que asignarle cada columna a una variable.

En el caso de la clase Rol al principio defini un usuario del tipo Usuario. Por lo que los datos obtenidos por el sp los voy a guardar en este objeto usuario. Para ello utilizo y defino los metodos DATAROWTOOBJECT (convertir fila en objeto basicamente).

public override void DataRowToObject(DataRow dr)

{

// Esto es tal cual lo devuelve el stored de la DB

this.rol\_id = Convert.ToInt32(dr["rol\_id"]);

this.Nombre = dr["rol\_nombre"].ToString();

this.Estado = Convert.ToBoolean(dr["rol\_estado"]);

}

Resumiendo. Si hago un **Entidad.FuncionSP(parametros, criterio**) me va a armar el nombre del storeProcedure como **funcionSPEntidadCriterio.** El cual tengo que definir en la base de datos.

Las funciones que tenemos son:

Guardar(parameterList) : insertEntidad

GuardarYObtenerID(parameterList) : insertEntidad\_RetornarID

Modificar(parameterList) : updateEntidad

Eliminar(parameterList) : deleteEntidad

Deshabilitar(parameterList) : deshabilitarEntidad

TraerListado(parameterList, condiciones):traerListadoEntidadCondiciones

TraerListado(condiciones) :traerListadoEntidadCondiciones

Ejemplos:

**unCliente.Guardar(unCliente.ParameterList)**

**sp: insertCliente**

Inserta el cliente que tenia en parameterList en la tabla Cliente

**Int idCliNuevo = unCliente.GuardarYObtenerID(unCliente.ParameterList)**

**Sp: insertCliente\_retornarID**

Inserta en la base en la tabla cliente un registro nuevo y me devuelve el idCliente que se le asigno a este nuevo cliente.

**Bool estado = unCliente.Modificar(unCliente.ParameterList)**

**Sp: updateCliente**

Intenta modificar un cliente según los parametros que le pase. Si lo logra me devuelve true, si no false. (puede pasar que los datos que le paso para modificar sean erroneos, o que se pise una fk, etc).

**unCliente.Eliminar(unCliente.ParameterList)**

**Sp: deleteCliente**

elimina un registro de la tabla Cliente. Habria que hacer que retorne un booleano para saber si se pudo efectuar o no.

**unUsuario.Deshabilitar(unUsuario.ParameterList)**

**Sp: deshabilitarUsuario**

me deshabilita un usuario, ya sea por que supero max intentos o problemas con las cuentas.

**DataSet ds = unUsuario.TraerListado(unUsuario.ParameterList, “DeRoles”)**

**sp: traerListadoUsuarioDeRoles**

Me trae un listado de los roles asociados que tiene un usuario. No olvidar que estos datos vienen en forma de tabla, igual que cuando vemos la consulta. Hay que hacer un datarowtoobject para obtener los campos asociados.

**DataSet ds = unBanco.TraerListado(“Completo”)**

**Sp: traerListadoBancoCompleto**

Me trae un listado de todos los bancos. La tabla banco completa. No le paso ningun parametro al sp.