```
using Entities;
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos
{
    public class DataCaja : DataConnection
    {
        #region Caja
        /*Acá encontramos otro de los tipos de métodos para el CRUD con sql.
         Este es te dipo DATASET porque necesitamos hacer un estilo de matriz
        para poder luego acomodar los datos de este dataset en un datagridview
        para que se puedan visualizar los datos encuadrados y ordenados*/
        public SaldosActualizados ConsultarSaldos(SaldosActualizados saldos)
        {
            /*Vamos a realizar la consulta correspondiente, siempre con la parametrización*/
            string query = @"exec sp_Calcular_Importes @Caja_ID";

            SqlParameter caja_ID = new SqlParameter("@Caja_ID", saldos.Caja_ID);

            SqlCommand cmd = new SqlCommand(query, conexion);
            cmd.Parameters.Add(caja_ID);

            try
            {
                OpenConnection();
                SqlDataReader reader = cmd.ExecuteReader();

                if (reader.Read())
                {
                    saldos.Importe_Inicial =
decimal.Parse(reader["Importe_Apertura"].ToString());
                    if (string.IsNullOrEmpty(reader["Ingreso"].ToString()))
                    {
                        saldos.Importe_Ingreso = 0;
                    }
                    else
                    {
                        saldos.Importe_Ingreso = decimal.Parse(reader["Ingreso"].ToString());
                    }
                    if (string.IsNullOrEmpty(reader["Egreso"].ToString()))
                    {
                        saldos.Importe_Egreso = 0;
                    }
                    else
                    {
                        saldos.Importe_Egreso = decimal.Parse(reader["Egreso"].ToString());
                    }
                    saldos.Total = saldos.Importe_Ingreso - saldos.Importe_Egreso +
saldos.Importe_Inicial;
                }
                reader.Close();
                cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
```

```
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }

            return saldos;
        }

        public int CerrarCaja(Cajas caja)
        {
            int resultado = -1;
            string query = @"exec sp_Cerrar_Caja @Empleado_ID_Cierre, @Fecha_Cierre,
@Importe_Cierre, @Importe_Cierre_Caja, @Caja_ID, @Caja_Abierta";

            SqlParameter estado = new SqlParameter("@Empleado_ID_Cierre",
caja.Empleado_ID_Cierre);
            SqlParameter fecha_Cierre = new SqlParameter("@Fecha_Cierre", caja.Fecha_Cierre);
            SqlParameter importe_Cierre = new SqlParameter("@Importe_Cierre",
caja.Importe_Cierre);
            SqlParameter importe_Cierre_Caja = new SqlParameter("@Importe_Cierre_Caja",
caja.Importe_Cierre_Caja);
            SqlParameter caja_ID = new SqlParameter("@Caja_ID", caja.Caja_ID);
            SqlParameter caja_Abierta = new SqlParameter("@Caja_Abierta", caja.Caja_Abierta);

            SqlCommand cmd = new SqlCommand(query, conexion);

            cmd.Parameters.Add(estado);
            cmd.Parameters.Add(fecha_Cierre);
            cmd.Parameters.Add(importe_Cierre);
            cmd.Parameters.Add(importe_Cierre_Caja);
            cmd.Parameters.Add(caja_ID);
            cmd.Parameters.Add(caja_Abierta);

            try
            {
                OpenConnection();
                resultado = cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return resultado;
        }

        public Cajas GetLastCajaID(Cajas caja)
        {
            string query = "exec sp_Get_Last_Caja_ID @Fecha";

            SqlParameter fecha = new SqlParameter("@Fecha", DateTime.Now);

            SqlCommand cmd = new SqlCommand(query, conexion);

            cmd.Parameters.Add(fecha);
```

```
            try
            {
                OpenConnection();
                SqlDataReader reader = cmd.ExecuteReader();

                if (reader.Read())
                {
                    if (string.IsNullOrEmpty(reader["Caja_ID"].ToString()))
                    {
                        caja.Caja_ID = -1;
                    }
                    else
                    {
                        caja.Caja_ID = int.Parse(reader["Caja_ID"].ToString());
                    }
                }
                reader.Close();
                cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return caja;
        }

        public bool CajaAbierta(DateTime Fecha)
        {
            bool cajaAbierta = false;

            string query = @"select Caja_Abierta from Cajas where Caja_ID = (select
max(Caja_ID) from Cajas)";

            SqlCommand cmd = new SqlCommand(query, conexion);

            try
            {
                OpenConnection();
                SqlDataReader reader = cmd.ExecuteReader();
                if (reader.Read())
                {
                    //Si el importe aparece, significa que la ultima caja abierta
                    //está cerrada
                    bool resultado = Convert.ToBoolean(reader["Caja_Abierta"]);
                    if (resultado)
                    {
                        cajaAbierta = true;
                    }
                    else
                    {
                        cajaAbierta = false;
                    }
                }
                reader.Close();
                cmd.ExecuteNonQuery();
```

```
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return cajaAbierta;
        }

        public int AbrirCaja(Cajas caja)
        {
            /*Este método es tipo int. Ya lo vimos antes.*/
            int resultado = -1;
            string query = @"exec sp_abrir_caja @Empleado_ID_Apertura, @Fecha,
@Importe_Apertura, @Caja_Abierta";

            SqlParameter empleado_Id = new SqlParameter("@Empleado_ID_Apertura",
caja.Empleado_ID_Apertura);
            SqlParameter fecha = new SqlParameter("@Fecha", caja.Fecha_Apertura);
            SqlParameter importe_Apertura = new SqlParameter("@Importe_Apertura",
caja.Importe_Apertura);
            SqlParameter caja_Abierta = new SqlParameter("@Caja_Abierta", caja.Caja_Abierta);

            SqlCommand cmd = new SqlCommand(query, conexion);

            cmd.Parameters.Add(empleado_Id);
            cmd.Parameters.Add(fecha);
            cmd.Parameters.Add(importe_Apertura);
            cmd.Parameters.Add(caja_Abierta);

            try
            {
                OpenConnection();
                resultado = cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return resultado;
        }
        public Cajas ConsultarIDCajaAbierta(Cajas caja)
        {
            /*Siempre que el método sea un objeto de la entidad, solo vamos a poder traer 1
entidad por consulta
             Por lo que, al no ser muchos datos o una lista. Lo único que traemos acá es la
consulta de 1 tabla
            Y solo 1 dato por propiedad de la entidad*/
            string query = @"select max(Caja_ID) as Caja_ID from Cajas";

            SqlCommand cmd = new SqlCommand(query, conexion);

            try
```

```
            {
                OpenConnection();
                /*Creamos un SQLDATAREADER que sirve para verificar si la consulta arroja
resultados o no
                 En caso de que si, y se puedan verificar, se los asignamos a la propiedad
correspondiente.
                En este caso, hemos solicitado 1 solo ID.*/
                SqlDataReader reader = cmd.ExecuteReader();
                if (reader.Read())
                {
                    caja.Caja_ID = int.Parse(reader["Caja_ID"].ToString());
                }

                //cerramos el reader y lo ejecutamos con el comando
                reader.Close();
                cmd.ExecuteReader();
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            //devolvemos el objeto.
            return caja;
        }

        public DataSet GetDetallesCajas(string buscar)
        {
            /*Acá hay otro DataSet
             Este lo creamos de tal forma que podamos recibir un argumento que va a servir
como parámetro de búsqueda
            Sería básicamente una manera de filtrar datos.
            Quitando eso, es lo mismo que el primero con un DataSet por lo que no cambia dsp
de la query*/
            string query;
            if (string.IsNullOrEmpty(buscar))
            {
                query = @"exec sp_get_cajas";
            }
            else
            {
                query = @"exec sp_Get_Cajas_Y_Detalles @Parametro";
            }

            SqlCommand cmd = new SqlCommand(query, conexion)
            {
                CommandType = CommandType.Text
            };

            cmd.Parameters.Add(new SqlParameter()
            {
                ParameterName = "@Parametro",
                SqlDbType = SqlDbType.NVarChar,
                Value = string.Format("%{0}%", buscar)
            });

            DataSet ds = new DataSet();
            SqlDataAdapter da = new SqlDataAdapter();
```

```
            try
            {
                conexion.Open();
                cmd.ExecuteNonQuery();
                da.SelectCommand = cmd;
                da.Fill(ds);
            }
            catch (Exception e)
            {
                throw new Exception("Error al listar las clases", e);
            }
            finally
            {
                conexion.Close();
                cmd.Dispose();
            }
            return ds;
        }

        public int RegistrarCobro(Detalles_Cajas detalles_Cajas)
        {
            int resultado = -1;
            string query = @"insert into Detalles_Cajas (Caja_ID,
                                                         Empleado_ID,
                                                         Importe_Ingreso,
                                                         Plan_Asignado_ID,
                                                         Observaciones)
                                                 values (@Caja_ID,
                                                         @Empleado_ID,
                                                         @Importe_Ingreso,
                                                         @Plan_Asignado_ID,
                                                         @Observaciones)"
            ;

            SqlParameter caja_ID = new SqlParameter("@Caja_ID", detalles_Cajas.Caja_ID);
            SqlParameter empleado_ID = new SqlParameter("@Empleado_ID",
detalles_Cajas.Empleado_ID);
            SqlParameter importe_Ingreso = new SqlParameter("@Importe_Ingreso",
detalles_Cajas.Importe_Ingreso);
            SqlParameter plan_Asignado_ID = new SqlParameter("@Plan_Asignado_ID",
detalles_Cajas.Plan_Asignado_ID);
            SqlParameter observaciones = new SqlParameter("@Observaciones",
detalles_Cajas.Observaciones);

            SqlCommand cmd = new SqlCommand(query, conexion);

            cmd.Parameters.Add(caja_ID);
            cmd.Parameters.Add(empleado_ID);
            cmd.Parameters.Add(importe_Ingreso);
            cmd.Parameters.Add(plan_Asignado_ID);
            cmd.Parameters.Add(observaciones);

            try
            {
                OpenConnection();
                resultado = cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
                throw;
            }
```

```
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return resultado;
        }

        public int RegistrarPago(Detalles_Cajas detalles_Cajas)
        {
            int resultado = -1;
            string query = @"insert into Detalles_Cajas (Caja_ID,
                                                         Empleado_ID,
                                                         Importe_Egreso,
                                                         Observaciones)
                                                 values (@Caja_ID,
                                                         @Empleado_ID,
                                                         @Importe_Egreso,
                                                         Observaciones)"
            ;

            SqlParameter caja_ID = new SqlParameter("@Caja_ID", detalles_Cajas.Caja_ID);
            SqlParameter empleado_ID = new SqlParameter("@Empleado_ID",
detalles_Cajas.Empleado_ID);
            SqlParameter importe_Egreso = new SqlParameter("@Importe_Egreso",
detalles_Cajas.Importe_Egreso);
            SqlParameter observaciones = new SqlParameter("@Observaciones",
detalles_Cajas.Observaciones);

            SqlCommand cmd = new SqlCommand(query, conexion);

            cmd.Parameters.Add(caja_ID);
            cmd.Parameters.Add(empleado_ID);
            cmd.Parameters.Add(importe_Egreso);
            cmd.Parameters.Add(observaciones);

            try
            {
                OpenConnection();
                resultado = cmd.ExecuteNonQuery();
            }
            catch (Exception)
            {
                throw;
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return resultado;
        }

        public DataSet GetDetallesDiarios(string buscar)
        {
            string query;
            if (string.IsNullOrEmpty(buscar))
            {
                query = @"exec sp_Detalle_Caja_Diario";
            }
            else
            {
```

```
                query = @"exec sp_Detalle_Caja_Diario_Con_Parametro @Parametro";
            }
            SqlCommand cmd = new SqlCommand(query, conexion)
            {
                CommandType = CommandType.Text
            };
            cmd.Parameters.Add(new SqlParameter()
            {
                ParameterName = "@Parametro",
                SqlDbType = SqlDbType.NVarChar,
                Value = string.Format("%{0}%", buscar)
            });

            DataSet ds = new DataSet();
            SqlDataAdapter da = new SqlDataAdapter();

            try
            {
                OpenConnection();
                cmd.ExecuteNonQuery();
                da.SelectCommand = cmd;
                da.Fill(ds);
            }
            catch (Exception e)
            {
                throw new Exception("Error al listar Detalles de Caja", e);
            }
            finally
            {
                CloseConnection();
                cmd.Dispose();
            }
            return ds;
        }



        #endregion
    }
}
```