public IDataReader ExecuteReader1 (string strUsername, int iUserId, string storedProcedureName)

{

IDataReader rdr = null;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

rdr = db.ExecuteReader(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

return rdr;

}

public IDataReader ExecuteReaderRun ()

{

IDataReader rdr = null;

string strOutParam = string.Empty;

string strOutParamValue = string.Empty;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand cmd = db.GetStoredProcCommand(storedProcedureName);

strOutParam = "@Param1";

db.AddOutParameter(cmd, strOutParam, param.ParamType, 255);

db.AddInParameter(cmd, "@Param2", param.ParamType, 23);

rdr = db.ExecuteReader(cmd);

strOutParamValue = Convert.ToString(db.GetParameterValue(cmd, strOutParam));

}

catch (SqlException ex)

{

throw ex;

}

outParam = strOutParamValue;

return rdr;

}

public DataTable RunStoredProcedureWithoutParametersNOSEUSA(string strUsername, int iUserId, string storedProcedureName)

{

DataTable dt = null;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

DataSet ds = db.ExecuteDataSet(dbcmd);

dt = ds.Tables[0];

}

catch (SqlException ex)

{

throw ex;

}

return dt;

}

public object ExecuteScalarObjectWithSQLQueryStringNOSEUA(string strUsername, int iUserId, string strSQLQuery)

{

object obj = null;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetSqlStringCommand(strSQLQuery);

obj = db.ExecuteScalar(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

return obj;

}

public Int32 ExecuteScalarIntegerWithStoredProcedureAndParametersNOSEUSA(string strUsername, int iUserId, string storedProcedureName, List<clsParameters> parameters)

{

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

if (parameters != null)

{

foreach (var param in parameters)

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, param.ParamValue);

}

}

object obj = db.ExecuteScalar(dbcmd);

if (obj == null || (obj != null && obj.Equals(DBNull.Value)))

{

return 0;

}

else

{

return Convert.ToInt32(obj);

}

}

catch (SqlException ex)

{

throw ex;

}

}

public IDataReader ExecuteReaderWithStoredProcedureAndParameters(string strUsername, int iUserId, string storedProcedureName, List<clsParameters> parameters)

{

IDataReader rdr = null;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

if (parameters != null)

{

foreach (var param in parameters)

{

if (param.ParamValue == null)

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, DBNull.Value);

}

else if (param.ParamType.Equals(DbType.DateTime))

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, Convert.ToDateTime(param.ParamValue));

}

else if (param.ParamType.Equals(DbType.Int32))

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, Convert.ToInt32(param.ParamValue));

}

else

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, param.ParamValue);

}

}

}

rdr = db.ExecuteReader(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

return rdr;

}

/// <summary>

/// This method takes sql query string and return IDataReader object.

/// </summary>

/// <param name="strSQLQuery"></param>

/// <returns></returns>

public IDataReader ExecuteReaderWithSQLQueryStringNOSEUSA(string strUsername, int iUserId, string strSQLQuery)

{

IDataReader rdr = null;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetSqlStringCommand(strSQLQuery);

rdr = db.ExecuteReader(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

return rdr;

}

/// <summary>

/// This method takes stored procedure name, input parameters and returns nothing

/// </summary>

/// <param name="storedProcedureName"></param>

/// <param name="parameters"></param>

public void ExecuteNonQueryWithStoredProcedureAndParameters(string strUsername, int iUserId, string storedProcedureName, List<clsParameters> parameters)

{

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

if (parameters != null)

{

foreach (var param in parameters)

{

if (param.ParamType.Equals(DbType.Binary))

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, param.ParamValueByteArray);

}

else

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, param.ParamValue);

}

}

}

db.ExecuteNonQuery(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// This method takes stored procedure name and returns nothing.

/// </summary>

/// <param name="storedProcedureName"></param>

public void ExecuteNonQueryWithStoredProcedureAndWithoutParametersNOSEUSA(string strUsername, int iUserId, string storedProcedureName)

{

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

db.ExecuteNonQuery(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

}

public void ExecuteNonQueryWithSQLQueryStringNOSEUSA(string strUsername, int iUserId, string strSQLQuery)

{

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetSqlStringCommand(strSQLQuery);

db.ExecuteNonQuery(dbcmd);

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

}

public void ExecuteNonQueryWithStoredProcedureAndInputAndOutputParametersNOSEUSA(string strUsername, int iUserId, string storedProcedureName, List<clsParameters> parameters, out string outParam)

{

string strOutParam = string.Empty;

string strOutParamValue = string.Empty;

try

{

Database db = factory.Create("DefaultConnection");

DbCommand dbcmd = db.GetStoredProcCommand(storedProcedureName);

if (parameters != null)

{

foreach (var param in parameters)

{

if (param.ParamDirection.Equals(ParamDirection.OUTPUT))

{

strOutParam = "@" + param.ParamName;

db.AddOutParameter(dbcmd, strOutParam, param.ParamType, 255);

}

else

{

db.AddInParameter(dbcmd, "@" + param.ParamName, param.ParamType, param.ParamValue);

}

}

}

db.ExecuteNonQuery(dbcmd);

strOutParamValue = Convert.ToString(db.GetParameterValue(dbcmd, strOutParam));

}

catch (SqlException ex)

{

throw ex;

}

catch (Exception ex)

{

throw ex;

}

outParam = strOutParamValue;

}

}