protected string MergeFields(List<ScheduleLoadEntity> scheduleReleaseListl, int totRows)

{

List < ScheduleLoadEntity > informacion = new DashboardBO().GetDIM\_IEX\_CT();

if (informacion.Count() == 0) return "No data in Database for Merge";

DateTime MINIMO = System.DateTime.Today.AddDays(100);

DateTime MAXIMO = System.DateTime.Today.AddDays(-100);

foreach (var row in scheduleReleaseListl)

{

if (row.SCHEDULE\_DATE > MAXIMO) MAXIMO = row.SCHEDULE\_DATE;

if (row.SCHEDULE\_DATE < MINIMO) MINIMO = row.SCHEDULE\_DATE;

}

string dia = MINIMO.ToString("ddd");

if (dia != "Sun") return "Start date " + MINIMO.ToDateString("mm/dd/yyyy") ;

dia = MAXIMO.ToString("ddd");

if (dia != "Sat") return "End date " + MINIMO.ToDateString("mm/dd/yyyy") ;

double rango = (MAXIMO - MINIMO).TotalDays;

= = = = = = = = = = = = = = = = = = = = = = =

LOOP  ITEMS one by one

BEGIN

        IF IT IS THE FIRST TIME OR THE INFORMATION FOR CT / SERVER / CUSTOMER IS DIFFERENT

               BEGIN

                           INSERT / UPDATE **SCHEDULE**  AND BRING BACK THE SCHEDULE ID

                           (USUALLY THIS WILL BE DONE 1 TIME IN THE WHOLE PROCESS, FILE IS NOT MIXING MULTIPLE SCHEDULES, ONLY 1 UNTIL NOW )

                 END

        INSERT / UPDATE  **PLAN\_SHRINK** FOR EACH ITEM

        (HERE IS WHERE THE PROCESS IS TAKING A LOT OF TIME)

END LOOP

//important step: merge the two lists excel and information

IEnumerable<Schedule> List2 = (from excel in scheduleReleaseListl

from info in informacion.Where(info =>

info.APP\_CODE == excel.APP\_CODE

&& info.CUSTOMER\_ID == excel.CUSTOMER\_ID

&& info.CT\_ID == excel.CT\_ID)

select new Schedule {

APP\_CODE = excel.APP\_CODE,

CUST\_ID = excel.CUSTOMER\_ID,

LOB = info.LOB

}).Distinct();

if (List2.Count() == 0)

{

return "Data could not be referred in database, check values in excel file for Server, Customer and CT\_ID ";

}

int LOB = 0;

int SK = 0;

int cust\_id = 0;

int CT\_ID = 0;

string server = "";

int next\_percent = 5;

//string usuario = UserSession.CurrentUser.EmployeeID;

string usuario = "90980";

try

{

usuario = UserSession.CurrentUser.EmployeeID;

}

catch(Exception ex)

{

usuario = "90980";

}

int schedule\_id = 0;

int conteo = 0;

int conteo2 = 0;

DataSet ds = new DataSet();

ds.Tables.Add("DS");

ds.Tables["DS"].Columns.Add("IEX\_SERVER");

ds.Tables["DS"].Columns.Add("CUSTOMER\_ID");

ds.Tables["DS"].Columns.Add("CT\_ID");

ds.Tables["DS"].Columns.Add("INTERVAL");

ds.Tables["DS"].Columns.Add("PLAN\_SHRINK");

//foreach (DataRow items in dtExcel.Rows)

foreach (var row in scheduleDetailsList2)

{

try

{

string p\_IEX\_SERVER = row.APP\_CODE;

int p\_CUSTOMER\_ID = row.CUSTOMER\_ID;

int p\_CT\_ID = row.CT\_ID;

DateTime p\_SCHEDULE\_RELEASE\_DATE = row.SCHEDULE\_RELEASE\_DATE;

DateTime p\_SCHEDULE\_DATE = row.SCHEDULE\_DATE;

int p\_INTERVAL = row.INTERVAL;

int p\_PLAN\_SHRINK = row.PLAN\_SHRINK;

if (server != p\_IEX\_SERVER || CT\_ID != p\_CT\_ID || cust\_id != p\_CUSTOMER\_ID)

{

server = p\_IEX\_SERVER;

CT\_ID = p\_CT\_ID;

cust\_id = p\_CUSTOMER\_ID;

//MASTER

schedule\_id = new DashboardBO().Load\_Schedule(LOB, p\_CT\_ID,

MINIMO, MAXIMO, p\_SCHEDULE\_RELEASE\_DATE,

usuario, p\_IEX\_SERVER, p\_CUSTOMER\_ID, SK);

bool ss2 = new DashboardBO().Load\_Schedule\_Plans(schedule\_id, usuario,

p\_INTERVAL, p\_PLAN\_SHRINK, p\_SCHEDULE\_DATE);

conteo++;

int porcen = conteo \* 100 / totRows;

if ( porcen >= next\_percent )

{

next\_percent = next\_percent + 5;

if (next\_percent > 100) next\_percent = 100;

string mensag1 = "Porcentage : " + porcen.ToString() + " ";

string ejecute = "<script type='text/javascript'>";

//ejecute = ejecute + "parent.updateProgress(" + porcen.ToString() + ");";

//ejecute = ejecute + " $('#cplMianDisplay\_Label1').val('"+ mensag1 +"' ); ";

ejecute = ejecute + " updateProgress("+porcen.ToString() + "); ";

ejecute = ejecute + "</script>";

//Response.Write(ejecute);

//Response.Flush();

string funcion = "updateProgress(" + porcen.ToString() + ");";

ScriptManager.RegisterStartupScript(this, GetType(), "HideSpin", funcion, true);

string function2 = "updateProgress(" + porcen.ToString() + ");";

ScriptManager.RegisterStartupScript(UpdatePanel1, UpdatePanel1.GetType(), "myFunction", function2, true);

}

}

catch(Exception ex)

{

conteo2++;

DataRow dr = ds.Tables[0].NewRow();

dr[0] = row.APP\_CODE;

dr[1] = row.CUSTOMER\_ID;

dr[2] = row.CT\_ID;

dr[3] = row.SCHEDULE\_RELEASE\_DATE;

dr[4] = row.SCHEDULE\_DATE;

dr[5] = row.INTERVAL;

dr[6] = row.INTERVAL;

//for (int j = 0; j < 7;j++ )

//{

// dr[j] = row[j].ToString();

//}

ds.Tables[0].Rows.Add(dr);

}

}

if (conteo2>0)

{

ShowExcel(ds.Tables[0]);

}

string fallo = ", failed records = "+ conteo2.ToString();

if (conteo2 == 0) fallo = "";

return "Loaded "+ conteo + " records " + fallo;

}

= = = = = = = = = = = = = = = = = = = = = = =

ANOTHER WAY

//important step: merge the two lists excel and information

var scheduleDetailsList2 = (from excel in scheduleReleaseListl

join info in informacion on new {excel.IEX\_SERVER,

excel.CUSTOMER\_ID,

excel.CT\_ID} equals

new {info.IEX\_SERVER,info.CUSTOMER\_ID,info.CT\_ID} into Result

select new ScheduleLoadEntity

{

CUSTOMER\_ID = excel.CUSTOMER\_ID,

CT\_ID = excel.CT\_ID,

IEX\_CT\_SK = Result.First() .IEX\_CT\_SK,

ECP\_LOB\_ID = Result.First().ECP\_LOB\_ID

}).Where(e=>e.ECP\_LOB\_ID > 0).Distinct();

if (scheduleDetailsList2.Count() == 0)

{

return "Data could not be referred in database, check values in excel file for Server, Customer and CT\_ID ";

}

var scheduleGrouped = scheduleDetailsList2.GroupBy(info => new

{

IEX\_SERVER = info.IEX\_SERVER,

CUSTOMER\_ID = info.CUSTOMER\_ID,

CT\_ID = info.CT\_ID,

});

bool success = false;

List<ScheduleLoadBase> failedRecords = new List<ScheduleLoadBase>();

foreach (var row in scheduleGrouped)

{

IList<ScheduleLoadEntity> rowsToUpload = null;

try

{

ScheduleLoadEntity entity = row.First();

rowsToUpload = row.ToList();

int schedule\_id = new DashboardBO().Load\_Schedule(entity.ECP\_LOB\_ID, row.Key.CT\_ID, MINIMO, MAXIMO,

entity.SCHEDULE\_RELEASE\_DATE, UserSession.CurrentUser.EmployeeID, row.Key.IEX\_SERVER, row.Key.CUSTOMER\_ID, entity.IEX\_CT\_SK);

success = new DashboardBO().Load\_Schedule\_Plans(schedule\_id, UserSession.CurrentUser.EmployeeID, rowsToUpload);

}

catch(Exception ex)

{

Common.DisplayMessage(this, "error", ExceptionLogging.LogException(ex));

return ex.ToString();

}

if(success == false && rowsToUpload != null)

{

failedRecords.AddRange(rowsToUpload.Select(rec => new ScheduleLoadBase

{

IEX\_SERVER = rec.IEX\_SERVER,

CUSTOMER\_ID = rec.CUSTOMER\_ID,

CT\_ID = rec.CT\_ID,

SCHEDULE\_RELEASE\_DATE = rec.SCHEDULE\_RELEASE\_DATE,

SCHEDULE\_DATE = rec.SCHEDULE\_DATE,

INTERVAL = rec.INTERVAL,

PLAN\_SHRINK = rec.PLAN\_SHRINK

}));

}

}

if (failedRecords.Count > 0)

{

ShowExcel(failedRecords);

}

return "Data Processing Completed";

}

protected void Button3\_Click(object sender, EventArgs e)

{

string url = HttpContext.Current.Request.Url.AbsoluteUri;

url = url.Replace("Load.aspx", ".aspx");

Response.Redirect(url);

}

}

Improvement

It goes for only the group

ScheduleLoadEntity entity = row.First();

int schedule\_id = new DashboardBO().Load\_Schedule(entity.ECP\_LOB\_ID, row.Key.CT\_ID, MINIMO, MAXIMO,

entity.SCHEDULE\_RELEASE\_DATE, UserSession.CurrentUser.EmployeeID,

row.Key.IEX\_SERVER, row.Key.CUSTOMER\_ID, entity.IEX\_CT\_SK);

It goes for ALL

rowsToUpload = row.ToList();

success = new DashboardBO().Load\_Schedule\_Plans(schedule\_id, UserSession.CurrentUser.EmployeeID, rowsToUpload);

public bool Load\_Schedule\_Plans(int schedule\_id, string p\_CREATED\_BY, IList<ScheduleLoadEntity> loadDetails)

{

return dashboardDAL.Load\_Schedule\_Plans(schedule\_id, p\_CREATED\_BY, loadDetails);

}

internal bool Load\_Schedule\_Plans( int schedule\_id, string CREATED\_BY, IList<ScheduleLoadEntity> **loadDetails**)

{

bool commit;

StringBuilder sbQuery = new StringBuilder();

sbQuery.AppendLine("**declare** v\_rowCount number; ");

sbQuery.AppendLine("**BEGIN** ");

sbQuery.AppendLine("**SELECT** count(\*) **INTO** v\_rowCount **FROM** CC\_CT\_PLAN\_SHRINK "

+" WHERE INTERVAL = :p\_INTERVAL AND SCHEDULE\_ID = :P\_SCHEDULE\_ID AND schedule\_date = :p\_SCHEDULE\_DATE; ");

sbQuery.AppendLine("IF v\_rowCount > 0 THEN ");

sbQuery.AppendLine("**UPDATE** CC\_CT\_PLAN\_SHRINK SET plan\_shrink = :p\_PLAN\_SHRINK, updated\_by = :p\_CREATED\_BY,updated\_on = SYSDATE");

sbQuery.AppendLine("**WHERE** INTERVAL = :p\_INTERVAL AND SCHEDULE\_ID = :P\_SCHEDULE\_ID AND schedule\_date = :p\_SCHEDULE\_DATE;");

sbQuery.AppendLine("**ELSE** ");

sbQuery.AppendLine("**INSERT** **INTO** CC\_CT\_PLAN\_SHRINK (PLAN\_SHRINK\_ID, SCHEDULE\_ID, INTERVAL, PLAN\_SHRINK, CREATED\_BY, CREATED\_ON, UPDATED\_BY, UPDATED\_ON, SCHEDULE\_DATE)");

sbQuery.AppendLine("**VALUES** (Cc\_ct\_plan\_shrink\_id\_seq.NEXTVAL, :P\_SCHEDULE\_ID, :p\_INTERVAL, :p\_PLAN\_SHRINK, :p\_CREATED\_BY, SYSDATE, :p\_CREATED\_BY, SYSDATE, :p\_SCHEDULE\_DATE);");

sbQuery.AppendLine("END IF;");

sbQuery.Append("END;");

// CREATE 2 ARRAYS, AND ASSIGN DEFAULT VALUES, schedule\_id and created by

int[] scheduleIDArray = new int[loadDetails.Count];

string[] createdByArray = new string[loadDetails.Count];

scheduleIDArray.InitializeAll(schedule\_id);

createdByArray.InitializeAll(CREATED\_BY);

// CREATE ARRAY WITH THE REST OF VALUES

int[] intervalArray = **loadDetails**.Select(rec => rec.INTERVAL).ToArray();

using (OracleCommand dbCommand = database.GetSqlStringCommand(sbQuery.ToString()))

{

dbCommand.ArrayBindCount = loadDetails.Count;

database.AddInParameter(dbCommand, ":p\_SCHEDULE\_ID", OracleDbType.Int32, scheduleIDArray);

database.AddInParameter(dbCommand, ":p\_CREATED\_BY", OracleDbType.Varchar2, createdByArray);

database.AddInParameter(dbCommand, ":p\_INTERVAL", OracleDbType.Int32, intervalArray);

database.AddInParameter(dbCommand, ":p\_PLAN\_SHRINK", OracleDbType.Decimal, **loadDetails**.Select(rec => rec.PLAN\_SHRINK).ToArray());

database.AddInParameter(dbCommand, ":p\_SCHEDULE\_DATE", OracleDbType.Date, **loadDetails**.Select(rec => rec.SCHEDULE\_DATE).ToArray());

using (OracleConnection connection = database.CreateConnection())

{

try

{

connection.Open();

database.ExecuteNonQuery(dbCommand, connection);

commit = true;

}

finally

{

database.CloseConnection(connection);

}

}

}

return commit;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Projection Metrics realted functions Ended\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}