**Steps run the monthly AFRS process**

The Jim Schiebe’s data is saved at Q:\Forecast206\DSHSRawData.

Step 1 – Run the package **AFRSDownloadFromDSHS.dtsx** (Jim Schiebe) . After run the task item “Import prior\_month\_sum (and special\_month\_sum if appropriate) into MonthlyOFMFile”, I could run the SQL statement to get the value for the “minus special rows” for the excel sheet.

* I create a table zxg.MonthlyOFMFile\_speical to store the special month data in the SSIS package AFRSDownloadFromDSHS\_SpecialFiles.dtsx.

Step 2 – Run the package **Assignments.dtsx,** this step provides the number of rows added to zxg.afrs\_abstract in the first time “Rows added 1st time through” in the excel file ***MonthlyCounts.xlsx***. This is in the task item **Lookup assignments for MonthlyOFMFile**

* I modified this package to add the cycle variable so that the afrs\_extract has new column Cycle, The value is provided by the SSIS variable of cycle in this package

Step 3 – Run the **UnassignedToExcel.dtsx.** This part populate the table zxg.BulkAssignmentsInsert using ZXG.MonthlyOFMFile using the very like svc\_id, meg\_id, allocation\_id. This process is based on the assumption that program index leads to Meg, subsubobject leads to service, allocation leads to source of fund.

* This package is modified to populate a table [zxg].[BulkAssignmentsInsertBeforeUpdate] with cycle variables to store the value before table BulkAssignmentsInsert is updated by 9999 for meg, 999 for svc and 9 for sof as not-forecasted cells. A view of [zxg].[vw\_AFRS\_BulkAssignmentsInsertBeforeUpdate] is also created.
* The column cycle in the table is provided by the variable cycle in this package.
* A table zxg.Afrs\_scheme\_assignments is created in this package in the first task item.

Step 4\* –This step is replaced with the tasked item in the step 3 above.

Step 5 – run the package **InsertCleanup.dtsx**.

Step 6 – run the **InsertBulkAssignments.dtsx**.: [zxg].[BulkAssignmentsInsert]🡪zxg.bulkassignments,this part update the table zxg.afrs\_xwalk, zxg.scheme, insert the new records into zxg.afrs\_assignments table.

Step 7-- run the package **InsertCleanup.dtsx**.

Step 8 – run the package **Assignments.dtsx.** This step creates “Rows added 2nd time through”

Step 9 – go to the table zxg.[afrs\_extract] and delete only the created date being today or whenever you ran the data.

Step 10 – repeat step 1 **AFRSDownloadFromDSHS.dtsx**

Step 11 – repeat step 2 **Assignments.dtsx**.

Step 11, run the package **RefreshTotalTransactions.dtsx.**

**Step 12, run the package UpdateTableAFRS\_TotalsByFiscalYear.dtsx after checking everything is good.**

How to use the proc [dbo].[spcExpenditureAuth\_Index\_Afrs\_Extract\_All] to check if there are missing codes

After all above done, we run the stored procedure like the following

exec zxg.spsFastTrack\_Validations\_All\_BudgetUnits '2017','15'

This stored procedure is part of the code in \\homedirs.eclient.wa.lcl\ofmhome$\xingguoZ\MyDocu\ForecastDocu\AFRS\SQL\_Code\ Process to Check the rows and expediture.sql.

**Steps run the permiminaryMonthlyData process**

Step 1 -- RunPreliminaryAssignments.dtsx

* Run only two task items (delete and load the data)

Step 2 - AFRSPreliminaryDownloadFromDSHS.dtsx

Step 3 - PreliminaryAssignments.dtsx: will get the value of for the cell “Rows added 1st time through”

Step 4 - UnassignedToExcel.dtsx. Reminder: Make change to anything that is MEG ’9999’ to Service to ‘999’ and SOF (source of fund) ‘9’ . This step is replaced with the tasked item in the step 3 above.

Step 5 – **InsertCleanup.dtsx**

Step 6 – **InsertBulkAssignments.dtsx**.

Step 7-- **InsertCleanup.dtsx**.

Step 8 – **PreliminaryAssignments.dtsx.**

Step 9 – RunPreliminaryAssignments.dtsx

Step 10 - AFRSPreliminaryDownloadFromDSHS.dtsx

Step 11 -- PreliminaryAssignments.dtsx

After all above done, we run the stored procedure like the following

exec zxg.spsFastTrack\_Validations\_All\_BudgetUnits\_Preliminary '2017','16'

Note: I checked with Eddy, that in the preliminary process, we will not do the missing code checking process as we usually do for the monthly process.

**After** run the above process and results are good, we need update the table dbo.afrs\_assignments and

dbo.AFRS\_Extract\_Preliminary from its counter parts under zxg schema.

**Run the BulkManualAdjustments.tdsx**

After we successfully executing this package and check the results are good, we then use table zxg.AFRS\_manualadjustments to update the table dbo.AFRS\_manualadjustments.

**Run the cfc\_scrub package UploadCFCEligiblesNew.dtsx**

**After executing the package, I need to run the following stored procedure:**

* exec zxg.spicfcScrub\_Eligibles\_fc 'FC1718'
* exec [zxg].[spicfcScrub\_Eligibles] @Scrub\_Month = '1715', this is to populate the table [zxg].[cfcScrub\_Eligibles]

Note:

After I run the monthly CFC data, I will run SAS code in the MainCycleProgram.sas:

* %ExecuteMethod(MirrorScrubEligTable.DataMgmt). before execution, make sure: %let SelScurbMonth=1910, i.e., set the scrub month to the correct month as shown in the example.

Notice usually after I run the monthly CFC and AFRS extract data, I should run the SAS code in the MainCycleProgram.sas:

* We do not run %ExecuteMethod(MirrorScrubEligFCTable.DataMgmt) (and above part codes in MainCycleProgram.sas**)**, since this is not monthly process, it is forecasting process, only run during the forecasting period for one time:
  + Usually not run %ExecuteMethod(CreateCycleViews.DataMgmt)
  + Usually not run %ExecuteMethod(PopulateForecastCells.Methods)
  + Usually not run %ExecuteMethod(UpdateForImports.Methods)
  + Usually not run %ExecuteMethod(MirrorScrubEligFCTable.DataMgmt), this is run only after I have run the scrub CFC FC eligible.
* we do not run A.04.05 - %ExecuteMethod(MirrorSqlServerDimTables.DataMgmt) because it was already run in the MasterStartFile.sas
* Run %ExecuteMethod(ExtractAfrsExpData\_Mon.PrMonthly): this process uses the view dbo.vw\_Afrs\_Extract in SQL server to create the SAS table MonEtl.\_ScrubSource.
  + not run %ExecuteMethod(ExtractAfrsExpData.PrMonthly) /\*for FC run with CO scrub\*/, this is only run when preparing for forecasting, since we do not apply the cost objective take out list (defined by the macro variable &Takeout1) for the history data and apply the cost objective takeout list to the data we are to use to produce the PT. Note: this is illustrated by the macro variable &CostObjScrubbOutDate which was used in creating table work.\_ExpIn1 in SAS code ExtractAfrsExpData\_190. This takeout action is to reconcile HCA’s manipulation on the expenditure data with their steps.
  + not run the %ExecuteMethod(ExtractAfrsExpData\_PrCy.PrMonthly) /\*for FC run with CO scrub and limit on AfrsCycMonth\*/, this is run when preparing for the forecast production and the data has to go back for earlier month for some reason.
* run %ExecuteMethod(TransformAfrsExpData.PrMonthly)
  + create table MonEtl.AfrsExpFinSum
  + update table MainDm.Map\_Afrs\_Expenditure
  + MONETL.\_SCRUBSOURCE --> Monetl.Afrsexpstaging--> MAINDM.MAP\_AFRS\_EXPENDITURE, MONETL.AFRSEXPSTAGING,etc.
* run %ExecuteMethod(LoadAfrsExpData.PrMonthly)
  + create table MonEtl.UnLagMonExp\_1910
  + populate table Maindm.Fact\_afrs\_expenditure
* run %ExecuteMethod(GenerateLagFactors.LagFactors)
* run %UpdateLagFactorsIn7206();/\*PFVer=PRELIM then LgProcLvl=pre else LgProcLvl=dat \*/
  + This part uses the table Lagfctr.LoadLag\_M and table dbo.Tracking\_Cycles in the SQL server to populate the table dbo.Tracking\_Lag\_Factors in the SQL server
* run %ExecuteMethod(LoadAfrsExpLagFctrs.PrMonthly)
  + insert/populate the table MainDm.Fact\_Afrs\_ExpenditureLag from tables in SQL server

Eddy,

For ExpAuthIndex of 5A2 and 582, please find the values for the following columns to populate table dbo.Cdtbl\_ExpAuthIndex:

* Cdtbl\_ExpAuthindex
* Cdtbl\_ExpAuthindex\_Desc
* Cdtbl\_ExpAuthindex\_CostAllocFund
* Cdtbl\_ExpAuthindex\_Fund\_Type
* Cdtbl\_ExpAuthIndex\_GFS\_Type
* Cdtbl\_Account\_Id
* Cdtbl\_FundType\_Id
* Cdtbl\_ExpAuthType\_Id

Thanks.

Xingguo