#functions that begin with "fn." are custom r functions

#this code does not need to be converted to Python, it is merely here as an FYI

#this code imports data from SQL into R and save the data in R ".r" data files

##########################~90 minutes

#install (tidyverse)

library(tidyverse) #select function https://www.datanovia.com/en/lessons/select-data-frame-columns-in-r/

##########################

ch <- odbcConnect("xpressfeed", uid = "xpressfeed", pwd = "xpressfeed")

setwd(r\_ri.num)

######################

date()

myquery = paste( "

SELECT s.\*

FROM vw\_co\_ss\_refview u1

LEFT JOIN vw\_sec\_sp500epsowned\_dvch\_dpmonsumm s

ON u1.gvkey = s.gvkey AND u1.primiss\_us\_iid\_mo = s.iid AND u1.dd\_af0 = s.dd\_af0

ORDER BY u1.dd\_af0, u1.gvkey , u1.primiss\_us\_iid\_mo

" , sep = '' )

temp <- sqlQuery(ch, stringsAsFactors = F , query = myquery )

dim(temp)

fn.asr( 'a0\_sec\_sp500epsowned\_dvch\_dpmonsumm\_ref', temp )

date()

rm(myquery)

#

###

# example - matches item number to mnemonic

#SELECT item\_number,  \* FROM xfl\_column WHERE  item\_number IN ('A21','A237','A25','A199','A237','A6','A60','A199','A237', 'A6', 'A25','A199','A181','A35','A10') #end example

#get data from xfl\_column table that tells us how to split adjust each data item

fn.asr('my\_xfl\_column', sqlQuery ( ch, " SELECT \* FROM xfl\_column " , stringsAsFactors = F ))

#download vw\_sec\_dvpxms\_thruexit

#download SQL tables as r dfs

#rewrite code below to also use names of sql views and then create SQL tables in SQL db

#myviews <- c( 'vw\_co\_ss\_refview' , 'vw\_co\_prccm5y\_nw\_pr', 'vw\_co\_prccm5y\_dvpxms') #'vw\_co\_afnd1\_ss' , 'vw\_co\_afnd2\_ss'

myviews <- c( 'vw\_co\_ss\_refview') #'vw\_co\_afnd1\_ss' , 'vw\_co\_afnd2\_ss'

mytables <- paste('a0\_', substr(myviews, 4, nchar(myviews) ), sep = '')

for ( i in 1:length(mytables) ) {

#sqlQuery(ch, stringsAsFactors = F , query = paste( 'DROP TABlE IF EXISTS ', mytables[i], sep = '') ) # drop sqltable

#sqlQuery(ch, stringsAsFactors = F , query = paste( 'SELECT \* INTO ', mytables[i], ' FROM ', myviews[i], sep = '') ) # create sqltable

# selectql table and save as r df

fn.asr(mytables[i], sqlQuery(ch, stringsAsFactors = F , query = paste( 'SELECT \* FROM ', myviews[i], ' ORDER BY dd\_af0, gvkey, primiss\_us\_iid\_mo', sep = '') ) )

} #end i

i

rm(myviews,mytables)

##############################################

#download annual dataframes, add new items to each df, create per share dfs, save dfs

#maybe re-write to first do all downloading, then all else (add new items to each df, save df; create per share dfs, save df; split adjust df's save dfs; etc.)

#define years for which we want data

mypers <- c(-2:5)

mypers

#combine a0\_co\_afnd1 & a0\_co\_afnd2, and lead or lag based on period i

# also adds new calculated items that we can later split adjust, calc cum sums of, etc.

for(p in 1: length(mypers)) {

i <- mypers[p]

#for each i period, get data in co\_adesind sql table

myquery = paste( "

SELECT af.\*

--SELECT u1.iid, u1.curcdm, u1.dateprc, af.\*

FROM vw\_co\_ss\_refview u1

LEFT JOIN co\_adesind af ON af.gvkey = u1.gvkey AND af.datadate = EOMONTH ( DATEADD ( year, " , i , ", u1.dd\_af0 ) ) AND af.CONSOL = 'C' and af.DATAFMT = 'STD' AND af.INDFMT='INDL' AND af.POPSRC='D'

ORDER BY u1.dd\_af0, u1.gvkey , u1.primiss\_us\_iid\_mo " , sep = '' )

mydata\_ad <- sqlQuery(ch, stringsAsFactors = F , query = myquery )

mytempname <- if( i < 0) paste( "mydata\_ad", '.m', abs(i), sep='') else paste("mydata\_ad", '.', abs(i), sep='')

#mytempname <- paste( "mydata\_ad", '.', i , sep='' )

fn.asr( mytempname , mydata\_ad )

rm(myquery , mydata\_ad) ; gc();gc()

#for each i period, get data in a0\_co\_afnd1\_ss sql table

rm(mydata\_af1, mydata\_af2, mydata\_af) ; gc();gc()

for(a in c(1,2) ) {

date()

#anntable <- paste ( 'a0\_co\_afnd' , a , '\_ss', sep = '' ) #'co\_afnd1\_ss' --'co\_afnd'

anntable <- paste ( 'co\_afnd' , a , sep = '' ) #'co\_afnd1\_ss' --'co\_afnd'

myquery = paste( "

SELECT af.\*

FROM vw\_co\_ss\_refview u1

LEFT JOIN " , anntable , " af ON af.gvkey = u1.gvkey AND af.datadate = EOMONTH ( DATEADD ( year, " , i , ", u1.dd\_af0 ) )

AND af.indfmt='indl' AND af.datafmt='std' AND af.consol='c' AND af.popsrc='d' --AND ad.curcd = 'usd'

ORDER BY u1.dd\_af0, u1.gvkey , u1.primiss\_us\_iid\_mo " , sep = '' )

#--LEFT JOIN co\_afnd1 af ON af.gvkey = u1.gvkey AND af.datadate = EOMONTH ( DATEADD ( year, -5 , u1.dd\_af0 ) )

#LEFT JOIN co\_afnd1 af ON af.gvkey = u1.gvkey AND af.datadate = EOMONTH ( DATEADD ( year, -5 , u1.dd\_af0 ) )

mydata\_af\_temp <- sqlQuery(ch, stringsAsFactors = F , query = myquery) #as.is = rep('numeric', nc) ) , as.is = c('numeric') , max = 0, buffsize = nr

date()

#

if (a==1) mydata\_af1 <- mydata\_af\_temp

if (a==2) mydata\_af2 <- mydata\_af\_temp

rm(anntable , myquery, mydata\_af\_temp ) ; gc();gc()

} #end a

a

colnames(mydata\_af1)[match("PacVerToFeedPop", colnames(mydata\_af1) )] <- 'PacVerToFeedPop1'

colnames(mydata\_af2)[match("PacVerToFeedPop", colnames(mydata\_af2) )] <- 'PacVerToFeedPop2'

st2 <- match ('popsrc', colnames (mydata\_af2) ) + 1

mycolnames\_af <- c ( paste ( 'co\_afnd1' , colnames (mydata\_af1) , sep = '.' ) , paste ( 'co\_afnd2' , colnames (mydata\_af2) [st2: ncol (mydata\_af2)] , sep = '.' ) )

if (p==1) fn.asr( 'mycolnames\_af' )

mycolnames\_af2 <- c ( colnames (mydata\_af1) , colnames (mydata\_af2) [st2: ncol (mydata\_af2)] )

if (p==1) fn.asr( 'mycolnames\_af2' )

#temp <- mydata\_af2 [st2: ncol (mydata\_af2)]

mydata\_af\_temp2 <- cbind ( mydata\_af1 , mydata\_af2 [st2: ncol (mydata\_af2)] )

#convert all columns in combined dataframe to numeric

mydata\_af <- data.frame( sapply(mydata\_af\_temp2, FUN= as.numeric)) #data.matrix(mydata\_af\_temp) #df2 = as.data.frame(sapply(df1, as.numeric)))

#

#add new items to mydata\_af here

#new items here

#

#add period number to colname in mydata\_af df below

mysuff <- if( i < 0) paste( ".m", abs(i), sep='') else paste('.', abs(i), sep='')

#mysuff <- paste('.', i, sep='')

#temp <- colnames(mydata\_af)

#colnames(mydata\_af) <- paste (temp , mysuff, sep='')

#save dataframes

tempfilename <- paste( "mydata\_af", mysuff, sep='')

fn.asr( tempfilename , mydata\_af )

rm(mydata\_af1, mydata\_af2, mydata\_af,mysuff, temp, tempfilename) ; gc();gc()

} # end i

#

p

i

a

#

#

rm(p,a,i)

#

#myunsure\_ps <- sqlQuery(ch, stringsAsFactors = F , query = "SELECT columnname, description, \* FROM xfl\_column

WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND (CHARINDEX('basic',description) = 0 AND CHARINDEX('dil',description) = 0) ORDER BY tablename, colnum")

#fn.asr('myunsure\_ps')

#mycols\_fd <- match( sqlQuery(ch, stringsAsFactors = F , query = "SELECT columnname FROM xfl\_column WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND CHARINDEX('dil',description) > 0 ORDER BY tablename, colnum")[,] , colnames(mydata\_af.0) )

#mycols\_pr <- match( sqlQuery(ch, stringsAsFactors = F , query = "SELECT columnname FROM xfl\_column WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND (CHARINDEX('basic',description) > 0 OR columnname IN ('opeps', 'bkvlps') ) ORDER BY tablename, colnum") [,] , colnames(mydata\_af.0))

#fn.asr('mycols\_pr')

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

#create list with information about columns in mydata\_af dataframes  
ch <- odbcConnect("xpressfeed", uid = "xpressfeed", pwd = "xpressfeed")

setwd(r\_ri.num)

fn.flga('mydata\_af.0')

#create vector of "unsure" per share items in co\_afnd1 and co\_afnd2 #manually add some of these to vectors below

co\_afnd\_all <- sqlQuery(ch, stringsAsFactors = F , query = "SELECT \* FROM xfl\_column

WHERE tablename IN ('co\_afnd1', 'co\_afnd2') ORDER BY tablename, colnum")

myunsure\_ps <- sqlQuery(ch, stringsAsFactors = F , query = "SELECT \* FROM xfl\_column

WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND (CHARINDEX('basic',description) = 0 AND CHARINDEX('dil',description) = 0) ORDER BY tablename, colnum")

#create vector of fully diluted or primary per share items in co\_afnd1 and co\_afnd2, then match to find colnums in co\_afnd tables that are fully diluted per share

mycols\_fd <- sqlQuery(ch, stringsAsFactors = F , query = "SELECT \* FROM xfl\_column WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND CHARINDEX('dil',description) > 0 ORDER BY tablename, colnum")

mycols\_pr <- sqlQuery(ch, stringsAsFactors = F , query = "SELECT \* FROM xfl\_column WHERE tablename IN ('co\_afnd1', 'co\_afnd2') AND adj\_method = '/' AND (CHARINDEX('basic',description) > 0 OR columnname IN ('opeps', 'bkvlps') ) ORDER BY tablename, colnum")

myshsinfo <- list( co\_afnd\_all = co\_afnd\_all, co\_afnd\_unsure = myunsure\_ps, co\_afnd\_fd = mycols\_fd, co\_afnd\_pr = mycols\_pr)

fn.as('myshsinfo')