```
/*

Oliver Seager
o.j.seager@lse.ac.uk
SE 16

This script essentially acts as an engine, importing and exporting datasets as is convenient for later code. It will work and the first of all 001?_data_processing scripts, alternating between Stata and Python as is appropriate.

Created: 02/12/2021
Last Modified: 02/12/2021

Infiles:
- cstat_jan1961_nov2021.dta (S&P Compustat Quarterly Fundamentals data. Fiscal 1961Q1-2021Q4, Calendar jan1961-nov2021)
- ffsurprises_1990_2009.xlsx (From Gürkaynak, Sack and Swanson (2005) via Nakamura and Steinsson (2018) (1990-1993) and Gorodnichenko and Weber (2015) (1994-2009). Data on FOMC-meeting FFF surprises. 1990-2009.)

Out&Infiles:


Outfiles:
- 001a_cstat_test.dta (Quarterly Compustat data with extraneous variables dropped and only observations from gvkey 005568 retained)
- 001a_cstat_qdates.dta (gvkey-datadate level Computstat data with Stata clock time endpoints of the current and previous fiscal quarter)
- 001a_ffsurprises_tctime.dta (Data on Federal Funds Rate surprises with Stata clock time timestamps of the time of the FOMC post-meeting statement release)

*/

************
* PREAMBLE *
************

clear all
set more off
macro drop _all
set rmsg on, permanently
capture log close
graph drop _all
set scheme modern, permanently

cd "C:/Users/Ollie/Dropbox/Monetary Policy and Innovation"

log using "./code/001a_data_processing.log", replace


**********************************************
* GETTING ALTERED VERSIONS OF COMPUSTAT DATA *
**********************************************

  ***********************
* Test Data for Compustat *
  ***********************

* Import Compustat *

use "./data/cstat_jan1961_nov2021.dta", clear


* Drop Observations Uniform for Single gvkey *

drop indfmt consol popsrc datafmt curcdq costat


```

```
* Drop Observations other than gvkey 005568 *

keep if gvkey == "005568"


* Sort on datadate *

sort datadate


* Export *

compress

save "./outputs/001a_cstat_test.dta", replace


  **********************************
* Compustat Quarterly Endpoints Only *
  **********************************

* Import Compustat *

use "./data/cstat_jan1961_nov2021.dta", clear


* Drop Extraneous Observations *

drop indfmt consol popsrc datafmt curcdq costat


* Add Current Quarter Endpoint *

rename datadate td_datadate // datadate comes in Stata's "date" format.

gen tc_current_q_end = cofd(td_datadate + 1) // Compustat lists data date as the last day of a given quarter. Hence, if a quarter ends a the end of december (1st January 00:00:00), we need to add on a day before converting to clock format. "Clock" time conversions are accurate to within a minute or so.

label var tc_current_q_end "Endpoint of Current Quarter in Clock Format"


* Add Lagged Quarter Endpoint *

gen tc_lag1_q_end = . // Initiate variable

label var tc_lag1_q_end "Endpoint of Previous Quarter in Clock Format"

sort gvkey fyear fqtr

by gvkey: replace tc_lag1_q_end = tc_current_q_end[_n-1] if ((fyear[_n-1] == fyear) & (fqtr >= 2)) | ((fyear[_n-1] == fyear - 1) & (fqtr == 1)) // Gets endpoint of previous quarter if previous quarter's data exists


* Adjust for Fiscal-time Accounting Changes and Data Errors, Generate Data Error Indicator *

by gvkey: gen current_lag1_diff = td_datadate - td_datadate[_n-1] if ((fyear[_n-1] == fyear) & (fqtr >= 2)) | ((fyear[_n-1] == fyear - 1) & (fqtr == 1)) // Gets length (in days) of current quarter if previous quarter's data exists

// Note that the shortest possible fiscal quarter is 89 days (For example, February 1st 2001 to April 30th 2001) and the longest possible fiscal quarter is 92 days (For example, June 1st to August 31st). Therefore, quarter lengths outside of this are indicative of accounting changes or data errors (Note that there are no observations outside of but within 5 days of this threshold, either side).

```

```
replace tc_lag1_q_end = . if current_lag1_diff < 89 | current_lag1_diff > 92 // Drop improper quarter lengths

drop current_lag1_diff // No longer needed

gen lagdate_error = 0 // Initiate error indicator

replace lagdate_error = 1 if tc_lag1_q_end == .

label var lagdate_error "Indicates Previous Quarter in Time Series not 3 Months Ago"


* Get "Is Leap Year" Indicator *

gen datadate_is_leap_year = 0 // Initiate indicator

replace datadate_is_leap_year = 1 if mod(yofd(td_datadate),4) == 0 // Replace if leap year

label var datadate_is_leap_year "Indicates Last Day of Quarter During Leap Year"


* Get Month of Year Variable *

gen datamonth = mod(mofd(td_datadate), 12) + 1 // Range 1-12

label var datamonth "1-12. Month of Last Day of Quarter"


* Get Day of Month Variable *

gen datadayofmonth = td_datadate - (dofm(mofd(td_datadate))) + 1 // Range 1-31

label var datadayofmonth "1-31. Day of Month of Last Day of Quarter."


* Manually Calculate Missing First Lagged Quarter Endpoints *

replace tc_lag1_q_end = tc_current_q_end - 89*(24*60*60*1000) if (tc_lag1_q_end == .) & (
datadate_is_leap_year == 0) & (datamonth == 5) & (datadayofmonth <= 28) // 89 day quarters
(Non-leap years only) (Last day of quarter 01/05-28/05)

replace tc_lag1_q_end = tc_current_q_end - 90*(24*60*60*1000) if (tc_lag1_q_end == .) & (
datadate_is_leap_year == 0) & (((datamonth >= 3) & (datamonth <= 4)) | ((datamonth == 5) & (
datadayofmonth == 29))) // 90 day quarters, non-leap years (Last day of quarter 01/03-30/04; 29/05)

replace tc_lag1_q_end = tc_current_q_end - 90*(24*60*60*1000) if (tc_lag1_q_end == .) & (
datadate_is_leap_year == 1) & (datamonth == 5) & (datadayofmonth <= 29) // 90 day quarters, leap
years (Last day of quarter 01/05-29/05)

replace tc_lag1_q_end = tc_current_q_end - 91*(24*60*60*1000) if (tc_lag1_q_end == .) & (
datadate_is_leap_year == 0) & (((datamonth == 5) & (datadayofmonth == 30)) | ((datamonth == 7) & (
datadayofmonth <= 30)) | ((datamonth == 12) & (datadayofmonth <= 30))) // 91 day quarters,
non-leap years (Last day of quarter 30/05; 01/07-30/07; 01/12-30/12)

replace tc_lag1_q_end = tc_current_q_end - 91*(24*60*60*1000) if (tc_lag1_q_end == .) & (
datadate_is_leap_year == 1) & (((datamonth >= 3) & (datamonth <= 4)) | ((datamonth == 5) & (
datadayofmonth == 30)) | ((datamonth == 7) & (datadayofmonth <= 30)) | ((datamonth == 12) & (
datadayofmonth <= 30))) // 91 day quarters, leap years (Last day of quarter 01/03-30/04; 30/05;
01/07-30/07; 01/12-30/12)

replace tc_lag1_q_end = tc_current_q_end - 92*(24*60*60*1000) if (tc_lag1_q_end == .) & (((
datamonth >= 1) & (datamonth <= 2)) | ((datamonth == 5) & (datadayofmonth == 31)) | (datamonth ==
6) | ((datamonth == 7) & (datadayofmonth == 31)) | ((datamonth >= 8) & (datamonth <= 11)) | ((
datamonth == 12) & (datadayofmonth == 31))) // 92 day quarters, all years (Last day of quarter
01/01-29/02; 31/05-30/06; 31/07-30/11; 31/12)


* Add Second Lagged Quarter Endpoint *
```

```

gen tc_lag2_q_end = . // Initiate variable

label var tc_lag2_q_end "Endpoint of Two Quarters Ago in Clock Format"

sort gvkey fyear fqtr

by gvkey: replace tc_lag2_q_end = tc_lag1_q_end[_n-1] if (lagdate_error == 0) & (lagdate_error[_n-1] == 0) & (((fyear[_n-1] == fyear) & (fqtr >= 2)) | ((fyear[_n-1] == fyear - 1) & (fqtr == 1))) // Gets endpoint of previous quarter if previous quarter's data exists


* Manually Calculate Missing Second Lagged Quarter Endpoints *

gen dog = tc_lag2_q_end

replace tc_lag2_q_end = tc_lag1_q_end - 89*(24*60*60*1000) if (tc_lag2_q_end == .) & (datadate_is_leap_year == 0) & (datamonth == 8) & (datadayofmonth <= 28) // 89 day quarters (Non-leap years only) (Last day of quarter 01/08-28/08)

replace tc_lag2_q_end = tc_lag1_q_end - 90*(24*60*60*1000) if (tc_lag2_q_end == .) & (datadate_is_leap_year == 0) & (((datamonth >= 6) & (datamonth <= 7)) | ((datamonth == 8) & (datadayofmonth == 29))) // 90 day quarters, non-leap years (Last day of quarter 01/06-31/07; 29/08)

replace tc_lag2_q_end = tc_lag1_q_end - 90*(24*60*60*1000) if (tc_lag2_q_end == .) & (datadate_is_leap_year == 1) & (datamonth == 8) & (datadayofmonth <= 29) // 90 day quarters, leap years (Last day of quarter 01/08-29/08)

replace tc_lag2_q_end = tc_lag1_q_end - 91*(24*60*60*1000) if (tc_lag2_q_end == .) & (datadate_is_leap_year == 0) & (((datamonth == 3) & (datadayofmonth <= 30)) | ((datamonth == 8) & (datadayofmonth == 30)) | ((datamonth == 10) & (datadayofmonth <= 30))) // 91 day quarters, non-leap years (Last day of quarter 01/03-30/03; 30/08; 01/10-30/10)

replace tc_lag2_q_end = tc_lag1_q_end - 91*(24*60*60*1000) if (tc_lag2_q_end == .) & (datadate_is_leap_year == 1) & (((datamonth == 3) & (datadayofmonth <= 30)) | ((datamonth >= 6) & (datamonth <= 7)) | ((datamonth == 8) & (datadayofmonth == 30)) | ((datamonth == 10) & (datadayofmonth <= 30))) // 91 day quarters, leap years (Last day of quarter 01/03-30/03; 01/06-31/07; 30/08; 01/10-30/10)

replace tc_lag2_q_end = tc_lag1_q_end - 92*(24*60*60*1000) if (tc_lag2_q_end == .) &  (((datamonth >= 1) & (datamonth <= 2)) | ((datamonth == 3) & (datadayofmonth == 31)) | ((datamonth >= 4) & (datamonth <= 5)) | ((datamonth == 8) & (datadayofmonth == 31)) | (datamonth == 9) | ((datamonth == 10) & (datadayofmonth == 31)) | ((datamonth >= 11) & (datamonth <= 12))) // 92 day quarters, all years (Last day of quarter 01/01-29/02; 31/03-31/05; 31/08-30/09; 31/10-31/12)


* Drop Extraneous Variables *

keep gvkey td_datadate tc_current_q_end tc_lag1_q_end tc_lag2_q_end // We keep gvkey-datadate to merge back in easily


* Export *

compress

save "./outputs/001a_cstat_qdates.dta", replace


*************************************************
* GETTING ALTERED VERSIONS OF FF SURPRISES DATA *
*************************************************

  ***************************************
* FF Surprises with Stata %tc Time Format *
  ***************************************

* Import Data *
```

```

import excel "./data/ffsurprises_1990_2009.xlsx", firstrow clear


* Get %tc Format Time *

gen tc_time = cofd(date) + time_milliseconds // Already calculated the number of milliseconds *into the day* in Excel

drop time_milliseconds // No longer needed


* Get Changes into Destringable Format *

// For some reason some of these variables contain an incorrect hyphen, which doesn't register in Stata as a minus sign

replace unexp_tw = subinstr(unexp_tw, "–", "-", .)

replace unexp_ww = subinstr(unexp_ww, "–", "-", .)

replace exp_tw = subinstr(exp_tw, "–", "-", .)

replace exp_ww = subinstr(exp_ww, "–", "-", .)

replace actual = subinstr(actual, "–", "-", .)


* Destring Changes *

destring unexp_tw unexp_ww exp_tw exp_ww actual, replace


* Export Data *

save "./outputs/001a_ffsurprises_tctime.dta", replace


*************
* POSTAMBLE *
*************

log close

exit
```