Conference Call Project Documentation (20211209)

Summary of Main Processing Pipeline

	Task	Codes	Input	Output
1	**Download Raw Data**			
1.1	Download Thomson One's Conference Calls [L]	mouse_key_recorder.py automatic_download .py	-	O1: [yyyymmdd-yyyymmdd].pdf O2: [yyyymmdd-yyyymmdd].xls
2	**PDF Processing**			
2.1	Convert Conference Calls from .pdf to .txt [M]	pdftransfer.sh	O1	O3: [yyyymmdd-yyyymmdd].txt
2.2	Split Conference Call .txt files to separate out individual conference calls, and combine with report information from .xls files [M, L]	ParseCCpdf.jl	O3, O2	O4: [yyyymmdd-yyyymmdd].csv
3	**Firm Identification (Firm Name Matching)**			
3.1	Download Compustat datasets [L]	-	-	O5: ciqcompany. sas7bdat O6: ciqcountrygeo. sas7bdat O7: wrds_gvkey. sas7bdat
3.2	Download Hassan dataset [L]	-	-	O8: Hassanfile_raw_ updated20219030.csv
3.3	Process Compustat and Hassan datasets into usable and truncated .csv files. [L]	convert_sas7bdattocsv.py hassanfilecsv_viewable_ truncate.py	O5, O6, O7, O8	O9: ciqcompany.csv O10: ciqcountrygeo.csv O11: wrds_gvkey.csv O12: Hassanfile_raw_ updated20219030 _truncated.csv
3.4	Match titles in conference calls with firm names in Hassan and Compustat datasets, with both exact and	Based on the current flow of work, stage 3 is technically done after stage 4 since it used entryfiles_combined, though the original	O9, O10, O11, O12	O13: CC_List[yyyy].csv O14: CC_List_2020-2021.csv

	fuzzy matching [M, L]	codes were written as stage 4 after stage 3.		
1	Identify keywords for the whole CC data.	CC_identify_keywords.py	CC Data on the server at "CriCount/group{X}" where X = 1, 2, 3 ..50	XXX1: CC Data with relevant keywords at "/project/kh_mercury_1/ CriCount/Full_Identified_ Keywords/group{X}" where X = 1, 2, 3 ..50
2	Concatenating all these files into a single dataset.	concatenateOutputs.py	XXX1	XXX2: Full_Master_Keywords.csv
3	Filter only those entries which were not collected during the first run	optimizedGetNewEntries.py	XXX2	XXX3: Full_New_Not_Done.csv
4	Filter based on a more exact keyword identification algorithm (rather than just checking in, doing a holistic check by looking at the spaces around the keyword)	getCorrect.py	XXX3	XXX4: Amended_Correct_No_IR.csv
5	Filter based on the presence of a percentage (the words percent, per cent, percentage, %) and then order based on the sorting rule provided by Kilian.	Ordering And Filtering.ipynb	XXX4	XXX5: Filtered_Ordered_Amended_ Correct_No_IR.csv
6	Perform the fuzzy matching between the Hassan/Compus	CCFuzzyMatch.ipynb	XXX5, Hassanfile_raw_ updated2019030_viewable.csv, ????: ciqcompany_merged withgvkeyandcountry.csv	XXX6: manual_full_updated_ conf_calls.xlsx

	tat and the CC datasets.			
4	**Keyword Identification**			
4.1	Make a folder structure with x groups (default: x = 50) [M, L]	mkdir.py dividefilesequallyinto folders.py	-	-
4.2	Make a list of keywords and template entry file [L]	-	-	O15: keyterms.txt O16: Entry mask.xlsx
4.3	Identify keywords in each conference call [M, L]	keyword_ident_1.py keyword_ident_1.sh	O15, O4	O17: FR5.csv
4.4	Extract all paragraphs from each conference call that contains a specific keyword [M, L]	keyword_ident_2.py keyword_ident_2.sh	O17	O18: TotalCircnew.xlsx
4.5	Cleans the identified matches and merges with gvkey dataset [L]	mergeclean.do	O18, O14	O19: cric1_newtotal.xlsx
4.6	Make a paragraph record file that splits the number of entries into groups of 500 [L]	make_paragraphrecord.py	O19	O20: paragraphrecord.xlsx
4.7	Bold the keywords and separate file into “entryfiles”, each containing 500 entries. [M, L]	makeentryfiles.py makeentryfiles.sh	O19, O20, O15, O16	O21: [i].xlsx
4.8	Combine entry files [L]	combine_entryfilesjason.py	O21	O22: entryfiles_ combined.xlsx

		combine_entryfilessixun.py combine_sixunandjasonentryfiles.py		
5	**Get Front Page Descriptions**			
5.1	Extract front page descriptions from conference calls [M, L]	extractdescriptioninfrontpage.py extractdescriptioninfrontpage.sh copyfiles.py copyfiles.sh	O21, O2, O3	O23: [yyyymmdd-yyyymmdd]_withfrontpagedesc.xlsx
5.2	Manually check through error cases and correct accordingly [L]	-	O23	O24: [yyyymmdd-yyyymmdd]_withfrontpagedesc.xlsx
5.3	Combine xls files [L]	combine_xlsfiles_withdescription.py	O24	O25: xlscombined_withfrontpagedescription.xlsx
5.4	Match and add front page descriptions to combined entry files [L]	-	O25, O22	O26: entryfiles_combined.xlsx (updated)

* M = Mercury, L = Local. [M] / [L] means this stage can be run on Mercury / locally (on your Booth Windows laptop) respectively. [M, L] means this stage can be run on both Mercury and your local laptop, where Mercury is preferred for large datasets and local is preferred for initial testing, debugging and small datasets.

1. Download Raw Data

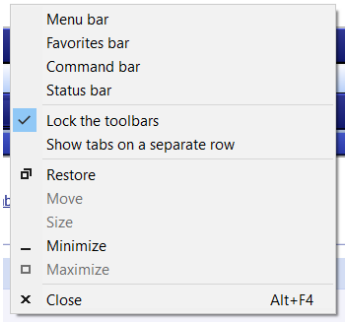
1.1 Download Thomson One's Conference Calls [L]

The goal is to download conference calls from Thomson One. This includes both the pdf files containing the actual calls, and xls files containing identifiers.

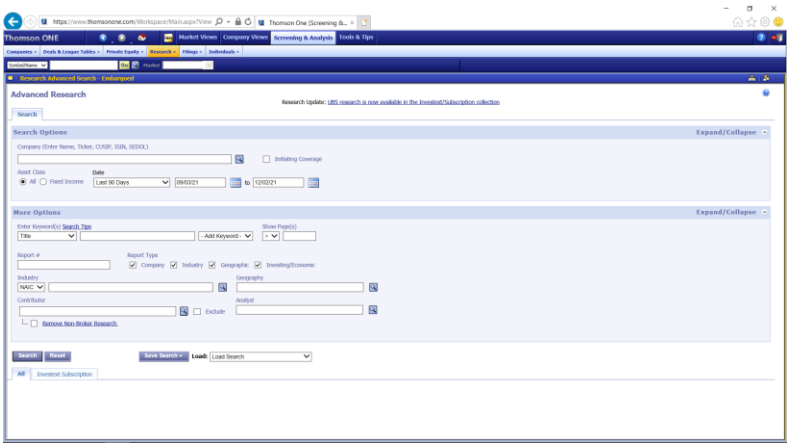
The main obstacles are that: (1) each page will only show 50 conference calls, (2) a maximum of 2,000 conference calls will be presented for every search, (3) web drivers are prohibited / web scraping doesn't work. The current solution is thus to write a python file that does auto-clicking, and this is automatic_download.py.

Before running the code:

- Open Internet Explorer (no other browers are allowed) to access Thomson One (proxy.uchicago.edu/login/thomsonone).
- Ensure that your browser settings are configured to enable the code to work. The goal is to hide away extraneous elements on the screen, so that no scrolling is needed to be able to click on all co-ordinates.
- The current code works for the Booth laptop, Lenono Thinkpad X1 Extreme (Windows 10) that is not connected to a HDMI screen. If it is connected to another screen, the current saved co-ordinates will likely be off. The screen resolution details (https://whatsmyscreensize.com/):
    - Screen Resolution
    - Width: 1920
    - Height: 1080
    - Device Pixel Ratio: 1.25
    - Display Dimensions (width x height): 16.0" x 9.0"
    - Screen Diagonal: 18.4" Screen
- The settings that worked for this set-up are:
    - Windows Taskbar: "Automatically hide the taskbar in desktop mode" is turned on
    - Internet Explorer settings: 125% zoom + the following



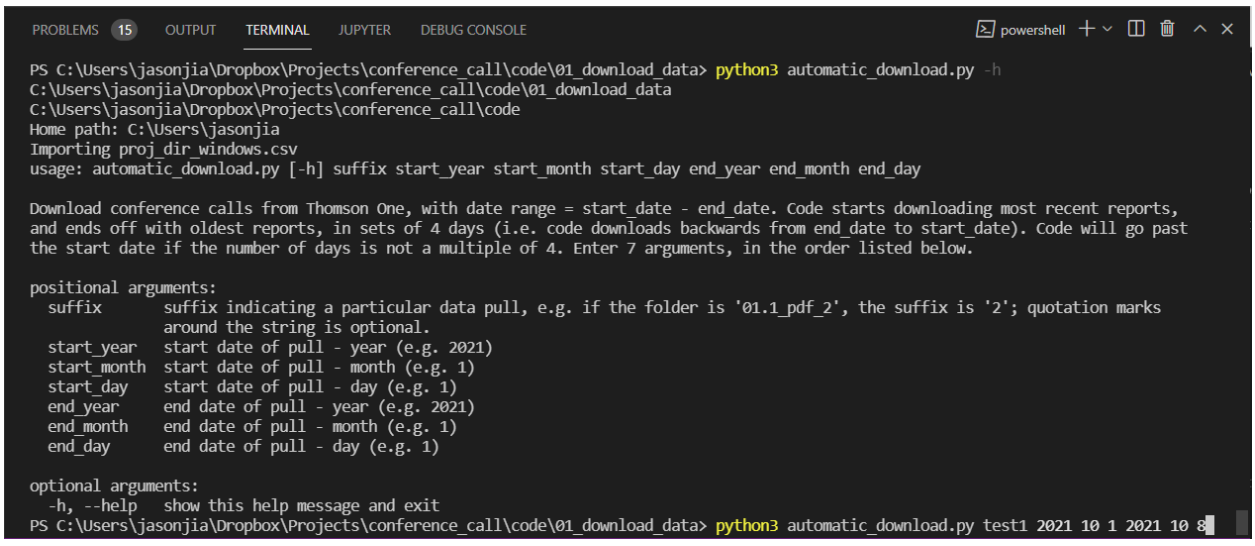
    - So that the screen looks like this:



- Create the necessary folders to store your output
    - Store the folders in: conference_call\output\01_download_cc
    - The root names are 01.1_pdf and 01.1_xls.
    - The suffix is used to separate different data pulls. Decide on a suffix.
    - Then create the two folders: “01.1_pdf_[suffix]” and “01.1_xls_[suffix]”.

To run the code:

- Open a terminal.
- Cd to “conference_call\code\01_download_data”.
- Run automatic_download.py, specifying the suffix, the start date (year, month, day) and the end date year, month, day):
    - E.g. python3 automatic_download.py test1 2021 10 1 2021 10 8

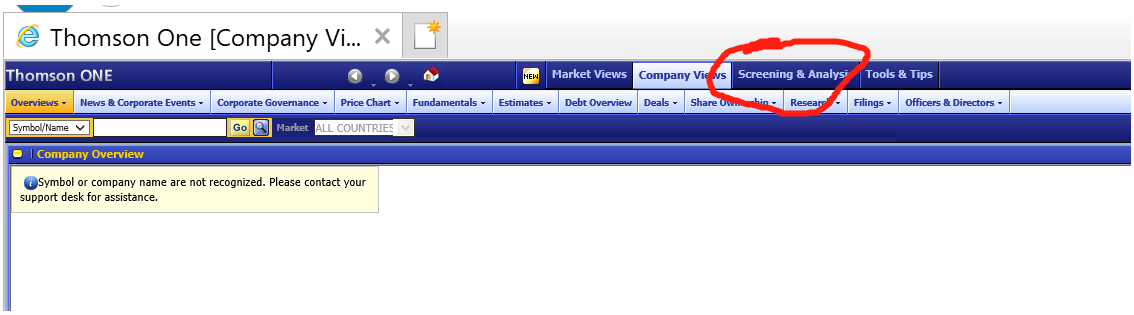


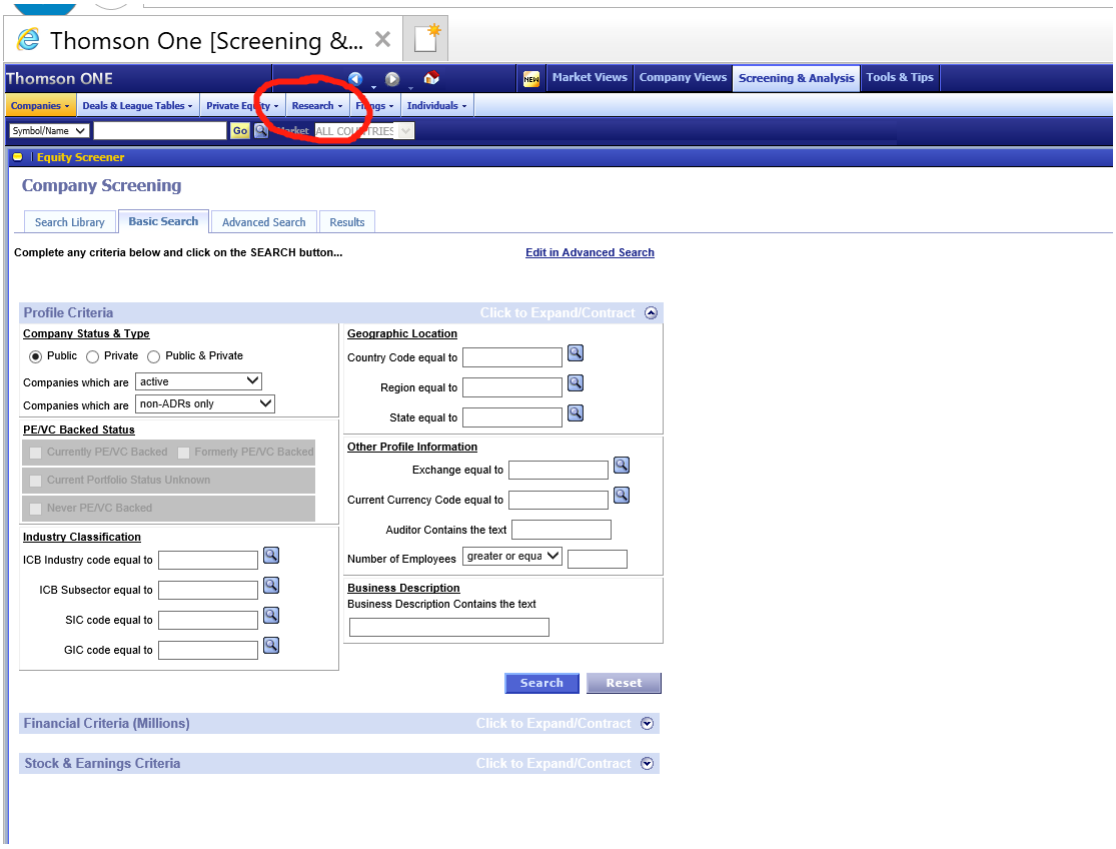
When running the code:

The main loop of the code tries to do the following:

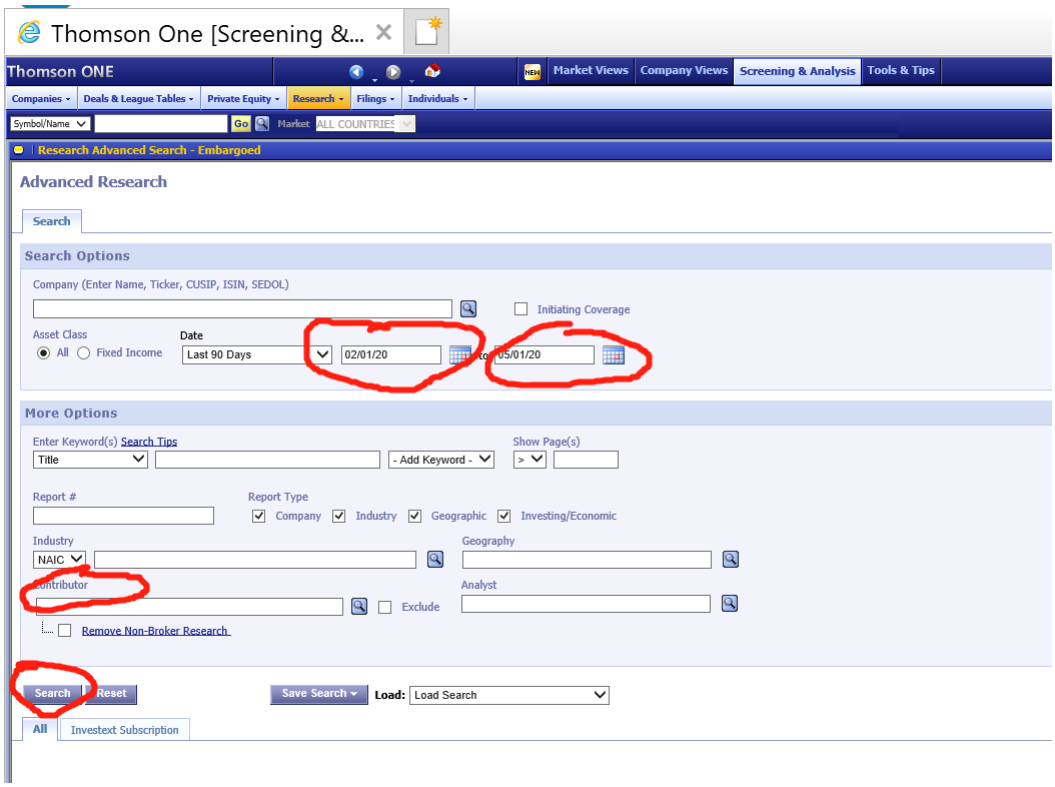
<u>(a) Enter search details</u>

- Click on Screening & Analysis -> Research



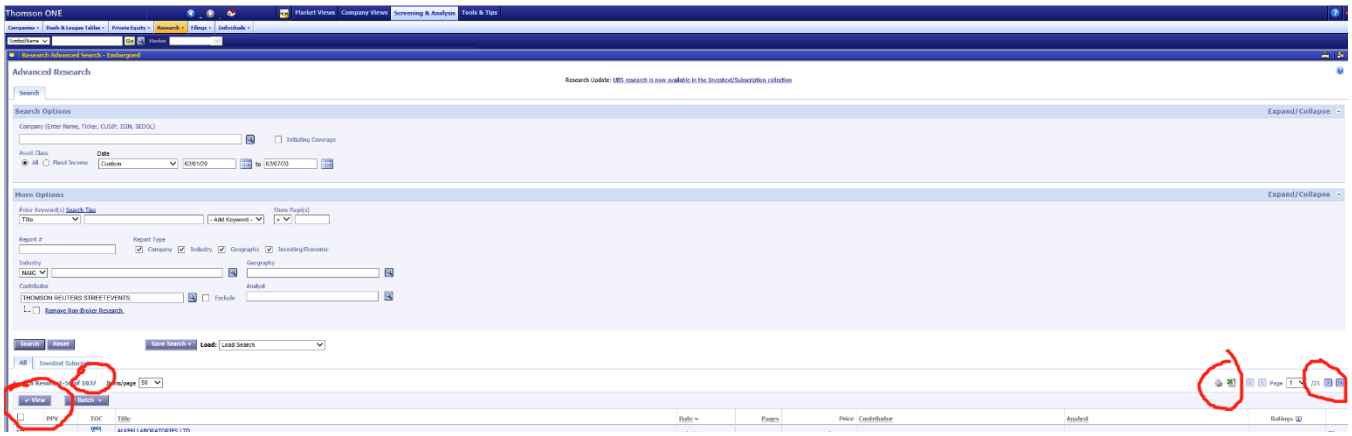


- Click on the Contributor field
- Type Streetevents in the Contributor field
- Click “Refinitiv Streetevents” in the drop-down list
    - Used to be “Thomson Reuter Streetevents” and may be something else in the future. More importantly, there should only be one option containing Streetevents, and it should be the correct option.
- Click on the start date field
- Type the start date
- Click on the end date field
- Type the end date
    - A 4-day time interval is used to ensure that each search gives no more than 2,000 calls.
- Press enter, which is equivalent to clicking the search button.



(b) Download the pdfs and xls files

- Copy the number of calls given by the search, save the number into a separate data set, and calculate the number of pages.
- Select all calls in one page
- Download the pdf file with all calls in that page into 01.1/pdf_[suffix]
- Download the xls file which contains information of each call into 01.1/xls_[suffix].
- To ensure that pdf and xls files are in pairs, check the existence of files in the folders.
- If there are some errors, the code will restart the process (by typing in the login websites, and automatically finishing the access steps.
- Uncheck all calls, click next page, repeat (b). If this is the last page of the date period, go back to (a) with date moving backwards.



Errors Handling

Errors can happen for many reasons, e.g. (1) automatic log off, (2) sudden network error, (3) system authentication error (log in failure), (4) change of file orders in the subsequent login, and (5) broken or corrupted files, and (6) unsuccessful download of files. The point is that some “manual coaxing” is necessary to help the code run smoothly from start to end.

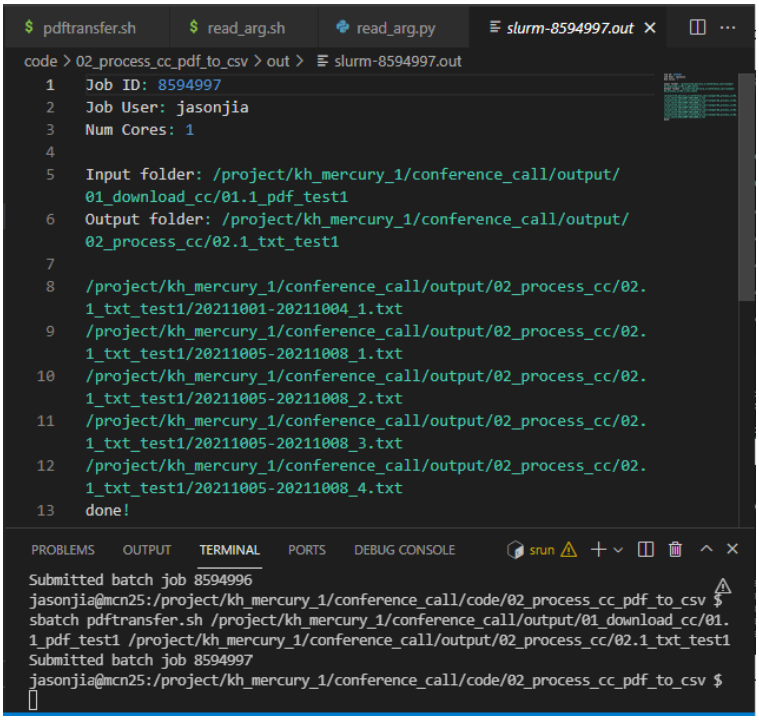
It is possible to try to account for all the errors, but from practical experience, the benefit of fewer errors, relative to the cost of more complex and harder-to-maintain code, diminishes quickly. We thus choose the following approach:

- If files fail to download, there is a time interval where the code will pause, for you to manually click to the correct state. Then, the code will try to save the file again.
- For all other errors, stop the code, get back into a workable state, and then rerun the code with a now truncated time frame.

2. PDF Processing

2.1 Convert Conference Calls from .pdf to .txt [M]

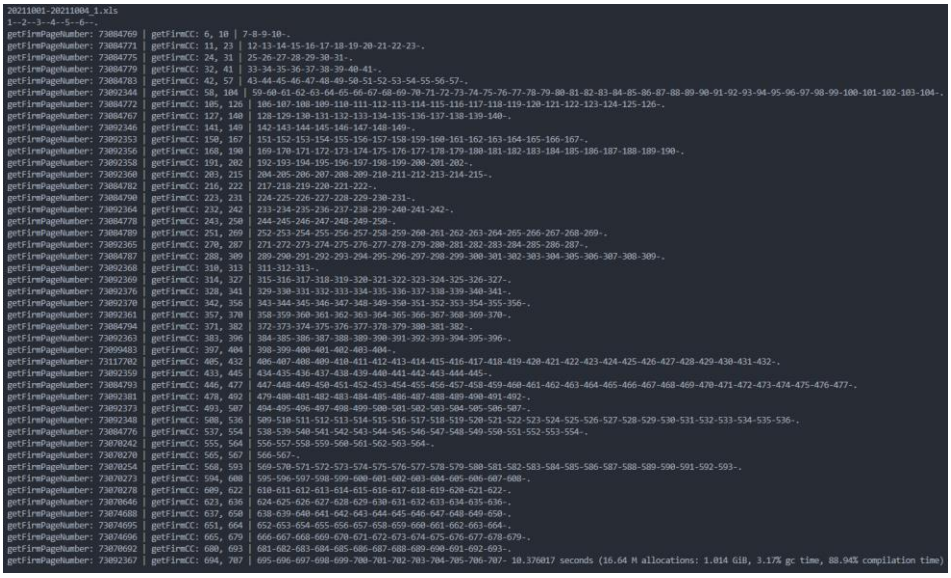
- Copy over the pdf and xls files to Mercury
- Run pdftransfer.sh on Mercury, specifying the full location of the input folder and the output folder
    - cd “conference_call/code/02_process_cc_pdf_to_csv”
    - sbatch pdftransfer.sh [input folder] [output folder]
- The important command is pdftotext, a Linux command that converts pdf to txt files. It also adds page and paragraph delimiters, which helps split the txt files by conference call later on.



- The txt files will be in the output folder

2.2 Split Conference Call .txt files to separate out individual conference calls, and combine with report information from .xls files [M, L]

- Copy over the txt files to Dropbox
- Run ParseCCpdf.jl
- Print messages have been added so you can see how the pdf files are processed. An example is given below:



- The output will be csv files that can be thought of as .xls files combined with the conference call text. This serves as the “primary database” containing the following variables: Title (firm name), Subtitle (firm name, date, and whether final/primary transcripts), Date, Pages (the number of pages of the call), Analyst (analysts who collect these transcripts, different analysts have slightly different forms of transcripts.), Report (Unique report number), Call (Raw call transcripts).
- The code:
    - Start from the delimiters contained in the .txt file to identify pages
    - Use title and pages in the information file to locate the beginning and end of each conference call.
    - Generate a new variable in the information file to store the raw call scripts.

3 Firm Identification (Firm Name Matching)

3.1 Download Compustat datasets [L]

- This step aims to match firms in conference calls to gvkeys, a unique firm identifier, as well as country information. In the xls (and csv) files, the 'title' variable gives the firm name associated with a particular conference call.
- Gvkeys are found in the Compustat – Capital IQ datasets. Access the database using Wharton Research Data Services (WRDS).

Steps:

- Register for an account and wait for approval by the IT team: https://wrds-www.wharton.upenn.edu/register/
- Sign into the SAS-studio web application: https://wrds-www.wharton.upenn.edu/pages/data/sasstudio-wrds/
- On the left, there is a folder directory, titled "Server Files and Folder".
- Capital-iq auxiliary files are located at wrds/capitaliq/sasdata/helper.
- Each table can be download by clicking the right mouse button. Although, it's not good practice, due to high size of tables.
- The best way to download data is to create Query (right mouse button on a table ⇒ new ⇒ Query).
- Downloading Query's result is a little bit tricky, since you can only print the result. The result is located in user's temporary folder. To find the name of the temporary folder: 1) open any table 2)push "display the code .." button 3) run the code 4) Result window → Engine/Host Dependent Information → filename shows your temporary folder.
- We used the following tables with specific columns from Compustat:
    - 1. CIQCOMPANY: companyid, companyname, tickersymbol, countryid and other columns.
    - Total: 24,511,757 obs. Public Companies: 66,256. Private Companies: 16,544,322. Public Investment Firms: 1,987 and Private Investment Firms: 203,407
    - 2. WRDS GVKEY: companyid, gvkey (115,357 observations).
    - 3. CIQCOUNTRY: countryid, countryname: countryid, countryname (221 countries).

3.2 Download Hassan dataset [L]

3.3 Process Compustat and Hassan datasets into usable and truncated .csv files. [L]

3.4 Match titles in conference calls with firm names in Hassan and Compustat datasets, with both exact and fuzzy matching [M, L]