This document describes the process of converting the C3S quality controlled file format (QFF) to the Common Data Model (CDM) file formatted tables.

QFF to CDMLite and QC tables.

The code is written in Python version 3.7.9.

Step 1

Set the working directory and output directories

OUTDIR2= "where the qc tables are to be saved"

OUTDIR = " where the cdmlite tables are to be saved "

os.chdir("where the QFF files are located")

Step 2

You have the option to use a list of required QFF files to be converted , which is useful, for running multiple jobs at once by chunking all the QFF files into txt lists of 5000.

my\_file = open("where your ls1.txt are located ", "r")

all\_filenames = my\_file.readlines()

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Or to process all the files at once in one job use:

extension = 'qff'

all\_filenames = [i for i in glob.glob('\*.{}'.format(extension))]

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Or to start the process from a specific last file use:

for filename in all\_filenames[all\_filenames.index('SWM00002338.qff'):] :

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Step 3

The code then proceeds to set up the master data frame (df) columns for the CDMlite. The code the extracts each hourly variable from the qff file and adds the details to the master df (e.g dft = temperature, dfdpt=dew point temp, dfslp = station level pressure, dfmslp = sea level pressure, dfwd = wind direction , dfws = wind speed). The code also sets up the qc table and adds the qc flags from the qff to a df (qct) for each variable. The code reads an external .csv file to a df2 and adds data policy and record numbers information to the variable df by merging the two df on primary\_station\_id, this is conducted on each variable df separately. External .csv can be found at: {<https://github.com/glamod/glamod-nuim/tree/master/PYTHON_CDM_Conversion_code/external_csv_files_for_conversion_tables_code>}

The code also conducts any value conversions to CDM requirements such as temperature in degrees Kelvin and pressure to pascals etc. and sets the values to the required decimal places.

Once this is completed for all variables the code merges all the variable df together into one {merged\_df} and all the variable qc df are merged into one df {qc\_merged\_df}. The code the converts the original qc flags to c3s flags base on the qc\_ definition tables. The final merged\_df and qc\_merged\_df are then saved as pipe separated files to the two separate out directories. The loop continues over all files until completed.

QFF to observations table.

The code is written in Python version 3.7.9.

Step 1

Set the working directory and output directories

OUTDIR = " where the cdmobs tables are to be saved "

os.chdir("where the QFF files are located")

Step 2

You have the option to use a list of required QFF files to be converted , which is useful, for running multiple jobs at once by chunking all the QFF files into txt lists of 5000.

my\_file = open("where your ls1.txt are located ", "r")

all\_filenames = my\_file.readlines()

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Or to process all the files at once in one job use:

extension = 'qff'

all\_filenames = [i for i in glob.glob('\*.{}'.format(extension))]

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Or to start the process from a specific last file use:

for filename in all\_filenames[all\_filenames.index('SWM00002338.qff'):] :

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Step 3

The code then proceeds to set up the master data frame (df) columns for the CDMobs. The code the extracts each hourly variable from the qff file and adds the details to the master df (e.g dft = temperature, dfdpt=dew point temp, dfslp = station level pressure, dfmslp = sea level pressure, dfwd = wind direction , dfws = wind speed). The code reads an external .csv file to a df2 and adds data policy and record numbers information to the variable df by merging the two df on primary\_station\_id, this is conducted on each variable df separately. External .csv can be found at: {<https://github.com/glamod/glamod-nuim/tree/master/PYTHON_CDM_Conversion_code/external_csv_files_for_conversion_tables_code>}

The code also conducts any value conversions to CDM requirements such as temperature in degrees Kelvin and pressure to pascals etc. and sets the values to the required decimal places.

Once this is completed for all variables the code merges all the variable df together into one {merged\_df} and all the variable qc df are merged into one df {qc\_merged\_df}. The final merged\_df are then saved as pipe separated files to the two separate out directories. The loop continues over all files until completed.

QFF to header table.

The code is written in Python version 3.7.9. The CDM header tables are created from the processed .psv observations tables.

Step 1

Set the working directory and output directories

OUTDIR = " where the cdmhead tables are to be saved "

os.chdir("where the cdmobs files are located")

Step 2

You have the option to use a list of required CDM obs files to be converted , which is useful, for running multiple jobs at once by chunking all the CDM obs files into txt lists of 5000.

my\_file = open("where your ls1.txt are located ", "r")

all\_filenames = my\_file.readlines()

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Or to process all the files at once in one job use:

extension = 'psv'

all\_filenames = [i for i in glob.glob('\*.{}'.format(extension))]

for filename in all\_filenames:

df=pd.read\_csv(filename, sep="|")

print(all\_filenames)

Step 3

The code reads the CDMobs files and only retains specific columns {cols\_list} into a df named merged\_df

{ col\_list = ["observation\_id", "report\_id", "longitude", "latitude", "source\_id","date\_time"]}

merged\_df=pd.read\_csv(filename, sep="|", usecols=col\_list)

The code then proceeds to set up the columns for the CDMhead {hdf} using some of the columns in the CDMobs {merged\_df}. The code reads an external .csv file to a df2 and adds required information to the variable hdf by merging the two df on primary\_station\_id. External .csv can be found at: {<https://github.com/glamod/glamod-nuim/tree/master/PYTHON_CDM_Conversion_code/external_csv_files_for_conversion_tables_code>}

The code also sets the values to the required decimal places. The code then remove sduplictae timestamp reports so that only one timestamp header report is now present.

hdf=hdf.drop\_duplicates(subset=['duplicates\_report'])

Once this is completed for all variables the CDMhead {hdf} are then saved as pipe separated files to the output directory. The loop continues over all files until completed.