This document describes the process of converting the GHCND daily .csv file to the Common Data Model (CDM) file formatted tables.

Daily.csv 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 .csv files are located")

Step 2

You have the option to use a list of required csv files to be converted , which is useful, for running multiple jobs at once by chunking all the csv 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 = 'csv'

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 filters the .csv file by only the rows of variables that a required for the observations table.

df = df[df["observed\_variable"].isin(["SNWD", "PRCP", "TMIN", "TMAX", "TAVG", "SNOW", "AWND", "AWDR", "WESD"])]

The code firstly creates the observations tables and qc definition tables form the csv files. The source flags are then converted to C3S source identifiers and the variables are converted to C3s variable identifiers. The code converts the variable values to CDM compliant values and ads the required CDM compliant elements relative to each variable such as value significance and units etc. The code converts the qc flags to either pass or fail. The code reads an external .csv file to a df2 and adds information such as data policy etc 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 creates the QC table {qct} and converts the GHCND qc flags to C3s qc flags for the qc\_definition table.

The code continues to create the header table from the observations table for each file.The code reads the observation df 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} and CDMobs {df} are then saved as pipe separated files to the output directories. The loop continues over all files until completed.