Documentation for
Level 1 data processing
Revised May 3, 2005

1.0 General Information

The purpose of the level 1 data processing programs is to read data from the 'raw' database tables (pecrsep and pedrsep), perform some basic quality control and data transformation, and then post it to the 'processed' database tables (pehpsep, peqpsep, and pedpsep). Each individual program is described in greater detail below. These programs are written in esql/C.

1.1 Enhancements/Bug Fixes/Changes

Build OB4

Enhancements

- Process_stage was changed to allow for Height data to be SHEF encoded.

Bug Fixes

For all level 1 processors

- all level 1 processors now use the sensok table properly
- how all the level 1 processors handle the data qualifier code "M' in the processed DB tables has been fixed
- the limitation on not allowing data older then 1990 has been removed
- the level 1 processors can now process up to one years' worth of data at a time

For process_precip

- The problems process_precip was having transforming the SHEF PE "PC" properly has been fixed

Build OB5

Enhancements

- Modified process_stage application to tabke into accound Build OB5 adb change to table rivercrit.
- Script, run_level1_process, used by oper's cron changed to use baseline amiirunning script to check to see if cron is still running this script from a previous execution time.

Build OB6

Bug Fixes

For all level 1 processors

- R1-36 Code changed so that a "/" at the end of the pathname defined by the apps defaults token *adb_pro_que* is not required.

For process_stage

- R1-40 Application modified so that negative stages can be handled.

1.2 Application Description

process_precip

This program reads 'raw' accumulated precipitation (SHEF PEDTSEP = 'PCIR*ZZ') from the pecrsep table and creates level 1 incremental precipitation to be stored in the 'processed' tables as follows:

- PPH – hourly increments (0z-1z, 1z-2z, etc.) -> pehpsep table
- PPQ – six hourly increments (0z-6z, 6z-12z, etc.) -> peqpsep table
- PPD – daily increments (12z-12z) -> pedpsep table

Quality control routines look at the entire time series and attempt to smooth out up/down fluctuations and try to recognize when a gage has been reset. There are very gross maximum allowed values defined within the program: MAXPPH=5, MAXPPQ=10, MAXPPD=20. It also looks in the sensok table for any pertinent entries.

transfer_precip

This program reads 'raw' incremental precipitation (SHEF PEDTSEP = 'PPHR*ZZ', 'PPQR*ZZ', 'PPDR*ZZ') from the pecrsep and pedrsep tables and creates level 1 incremental precipitation to be stored in the 'processed' tables as follows:

- PPH – from pecrsep -> pehpsep table
- PPQ – from pecrsep -> peqpsep table
- PPD – from pedrsep -> pedpsep table

**NOTE: PPH & PPQ transfer one less day of data than PPD because the first day of data is used as a buffer for finding the 0z value.

Quality control routines read in the value of the quality_code column from the pedrsep table. There are very gross maximum allowed values defined within the program: MAXPPH=5, MAXPPQ=10, MAXPPD=20. It also looks in the sensok table for any pertinent entries.

process_stage

This program reads 'raw' instantaneous height (i.e. stage, pool) and flow (SHEF PEDTSEP = 'H*IR*ZZ', 'Q*IR*ZZ') from the pecrsep table and creates level 1

hourly instantaneous height, flow, and storage to be stored in the pehpsep table as follows:

- H* -> all transferred to level 1 H*
    - If *=G and rating exists -> also process to level 1 QR
    - If *=P and rating exists -> also process to level 1 LS
- Q* -> all transferred to level 1 Q*

Quality control routines use information from the rivercrit table to determine the maximum and minimum flow/height values, as well as the maximum rate of change in flow/height, for each site. Sites for which flow is computed from height have each time series quality controlled separately with the rivercrit values so it is possible that their quality flags may not match. It also looks in the sensok table for any pertinent entries for the 'raw' SHEF PEDTSEP and applies quality flags the same for all processed time series.

process_temp

This program reads 'raw' instantaneous temperature (SHEF PEDTSEP = 'TAIR*ZZ') from the pecrsep table and creates level 1 hourly instantaneous temperature to be stored in the pehpsep table. It also creates level 1 maximum and minimum temperature to be stored in the pedpsep table.
**NOTE: max/min values determined through this program will not overwrite max/min values from the transfer_txn program (see below).

Quality control routines look at the entire time series and attempt to eliminate unreasonable jumps between readings. There are very gross maximum and minimum allowed values defined within the program: MAX=130, MIN=-50. It also looks in the sensok table for any pertinent entries.

transfer_txn

This program reads 'raw' daily maximum and minimum temperature (SHEF PEDTSEP = 'TAIR*XZ', 'TAIR*NZ') from the pedrsep table and creates level 1 daily maximum and minimum temperature to be stored in the pedpsep table.
**NOTE: max/min values from this program take precedence over those determined through process_temp by using a SHEF data qualifier code of 'V'.

Quality control routines read in the value of the quality_code column from the pedrsep table. There are very gross maximum and minimum allowed values defined within the program: MAX=130, MIN=-50. It also looks in the sensok table for any pertinent entries.

process_sw

This program reads 'raw' instantaneous snow water equivalent (SHEF PEDTSEP = 'SWIR*ZZ') from the pecrsep table and creates level 1 daily 12z instantaneous snow water equivalent to be stored in the pedpsep table.

Quality control routines look at the entire time series and attempt to eliminate unreasonable jumps between readings. There are very gross maximum and

minimum allowed values defined within the program: MAX=500, MIN=0. It also looks in the sensok table for any pertinent entries.

process_flow

This program reads 'raw' daily flow (SHEF PEDTSEP = 'Q*DR*ZZ') from the pedrsep table and creates level 1 daily flow to be stored in the pedpsep table.

Quality control routines read in the value of the quality_code column from the pedrsep table. The program screens out negative values. It also looks in the sensok table for any pertinent entries.

1.3 Design Considerations

The SHEF data qualifier codes that are used by the programs are based on SHEF Version 2.0.

Any data that is read in from the pecrsep and pedrsep tables with a SHEF qualifier code that indicates a 'good' value (G, M, S, V, P) will not be set bad in these programs as it is assumed to have been through some other quality control. Data that is read in with a qualifier code that indicates a questionable value (F, Q) will be treated as lower quality data than values with 'good' or 'unspecified' codes when being compared within the time series, but may still pass the level 1 quality control procedures. Data that is read in with a qualifier code that indicates a 'bad' value (B, R) will not be set good in these programs.

The quality_code column in the pedrsep table is set during posting using the locdatalimits and datalimits tables (see IHFS Quality Code Operations Guide for more information). This code is taken into account before these programs perform their quality control procedures so that values which did not pass this initial screening will not be set good through the level 1 processing routines.

SHEF data qualifier codes for the values output from these programs are as follows:

- R = rejected; did not pass level 1 or was set bad in a previous quality control method
- F = flagged; set bad through entries in the sensok table
- S = screened; passed level 1 (exception: not used by transfer_txn program)
- V = verified; passed level 1 at a higher level than 'S' (only used by transfer_txn program)

2.0 Configuration Information

These programs make use of the following apps_defaults tokens:

adb_name	archive database name
adb_dir	archive base directory
adb_pro_que	base directory for 'processed' poster

proclev1	optionally defined token for the SHEF parameter 'type' code of level 1 processed data; default is '1'

These programs are set to run on the cron every ten minutes. They determine which stations to process each time by using the new_report flag in the ingestfilter table, which is set to 'Y' when a new value is posted and then set to 'N' after the level 1 processing program has run.

Log files are created in $(adb_dir)/logs/process and overwritten each run.

Shef data files are created in $(adb_dir)/data/process before being copied to the processed posting queue (with a time stamp) and are overwritten each run.

3.0 User How-To

These programs may be run manually either through the *arcmenu > Data Processing menu*, or on the command line by simply typing the program name. It is a good idea to run a different version of the program than that which is on the cron, otherwise there is a chance that the output files could clash. There are four different 'versions' of the programs, determined through user input, that generate different output file names.

All programs allow for optional command line arguments that control the 'test' and 'debug' modes (these must be entered before any other arguments):

-t = test mode; do not write to database
-d = debug mode; write everything to the log file

All programs also allow for user inputs to be entered either on the command line or when prompted within the program. The cron programs use the 'default' command line option, which uses default values as described below. Following is a short description of the inputs for each program.

process_precip

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named proc_precip.log, ppd_proc.shef, ppq_proc.shef, pph_proc.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all

1) all active entries with PEDTSEP=PCIR*ZZ
2) output file names now become *.all

ii. new **- default**

1) only those entries from above with new_report='Y'

b. input file name

3. type of processing
    a. all = create 24, 6, and 1 hour increments **- default**
    b. dly = create 24 hour increments only
    c. six = create 6 hour increments only
    d. hrly = create 1 hour increments only
    e. dlysix = create 24 and 6 hour increments
    f. sixhrly = create 6 and 1 hour increments
    g. dlyhrly = create 24 and 1 hour increments
4. start date for processing (YYYYMMDD)
    a. **default is 3 days back**
5. end date for processing (YYYYMMDD)
    a. **default is today**

transfer_precip

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named trans_precip.log, ppd_trans.shef, ppq_trans.shef, pph_trans.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=PPDR*ZZ, PPQR*ZZ, or PPHR*ZZ
            2) output file names now become *.all
        ii. new **- default**
            1) only those entries from above with new_report='Y'
    b. input file name
3. start date for processing (YYYYMMDD)
    a. **default is 5 days back**
4. end date for processing (YYYYMMDD)
    a. **default is today**

process_stage

1. *optional*

a. -high = use the ‘highscreenf’ value from the rivercrit table to throw out high flows
b. default is to use the ‘ultimatescreenf’ (or ‘damscreenf’) value

2. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named proc_stage.log, h_proc.shef, h_trans.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
3. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=H*IR*ZZ or Q*IR*ZZ
            2) output file names now become *.all
        ii. new **- default**
            1) only those entries from above with new_report=’Y’
    b. input file name
4. start date for processing (YYYYMMDD)
    a. **default is 2 days back**
5. end date for processing (YYYYMMDD)
    a. **default is today**

process_temp

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named proc_temp.log, ta_proc.shef, txn_proc.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=TAIR*ZZ
            2) output file names now become *.all
        ii. new **- default**
            1) only those entries from above with new_report=’Y’
    b. input file name

3. start date for processing (YYYYMMDD)
    a. **default is 1 days back**
4. end date for processing (YYYYMMDD)
    a. **default is today**

transfer_txn

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named trans_txn.log, txn_trans.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=TAIR*(X/N)Z
            2) output file names now become *.all
        ii. new **- default**
            1) only those entries from above with new_report='Y'
    b. input file name
3. start date for processing (YYYYMMDD)
    a. **default is 1 days back**
4. end date for processing (YYYYMMDD)
    a. **default is today**

process_sw

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named proc_sw.log, sw_proc.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=SWIR*ZZ
            2) output file names now become *.all
        ii. new **- default**

1) only those entries from above with new_report='Y'

b. input file name

3. start date for processing (YYYYMMDD)
    a. **default is 1 days back**
4. end date for processing (YYYYMMDD)
    a. **default is today**

process_flow

1. source for list of stations to process
    a. db
        i. create list from ingestfilter table **- default**
        ii. output files named proc_flow.log, q_proc.shef
    b. file
        i. read list from file (with format LID PEDTSEP)
        ii. output files named same as in (a) with .file extension
    c. default
        i. use default values for all inputs
        ii. output files named same as in (a) with .def extension (used by cron)
2. based on source from above
    a. which ingestfilter stations to process
        i. all
            1) all active entries with PEDTSEP=Q*DR*ZZ
            2) output file names now become *.all
        ii. new **- default**
            1) only those entries from above with new_report='Y'
    b. input file name
3. start date for processing (YYYYMMDD)
    a. **default is 5 days back**
4. end date for processing (YYYYMMDD)
    a. **default is today**

4.0 Troubleshooting Information

If there are concerns with how a site's data is being processed, run the specific processing program with the debug and test modes turned on and with input from a file and look at the log and data files. If the user still has problems, contact the RFC Support Group.

5.0 Maintenance Information

Originating Programmer/Office: Alcorn, Brenda
Colorado Basin River Forecast Center
Salt Lake City, UT

Maintenance Programmer/Office: Alcorn, Brenda
Colorado Basin River Forecast Center

Salt Lake City, UT

6.0 References

Archive Database data dictionary
SHEF Version 2.0 Handbook
IHFS Quality Code Operations Guide