Alpha Report

End of august 2018

Alpha Deliverables for the software:

1: Have a database capable of containing all data needed by research group. Database can be accessed, queried for individual data points across any time range where those exist, data can then be exported into a csv for processing.

2: Build reader in python with the following requirements: Can inventory existing datasets, can tag datasets by type, time and geographical location, can format data sets into standardized structures.

3. Build processor to properly organize data pulled from the reader into the database, while maintaining the integrity of the data and preventing loss through sql injection, corruption of data, or other issues with input

Tests for completion (TFC) based on these deliverables:

1 Database

- 1.1 Database is fully accessible.
- 1.2 Database queries provide accurate outputs.
- 1.3 Outputs can be exported into usable files.

2 CSV reader

- 2.1 Program accurately interprets csv column data types (data, record number, max temp...etc)
- 2.2 Program formats data to parseable state (checks for inconsistencies in the data)
- 2.3 Program pulls headers from CSV and converts the strings into formats that SQL can use for its own headers

2.4 Cleanses any damaging inputs (prevention of sql injection)

3 CSV-SQL Builder

3.1 Program creates Table for each Geolocation (eg “Canada_Creek”) based on UI query

3.2 Populates tables with safe, escaped, and standardized data from CSV

3.3 Output line number confirmation to ensure all data input is audited

Development QA results:

Testing/Datapath process:

Loading data into database

- Using AWS data from Canada Creek, began loading of raw 5 min data into the automated reader.
- Reader began processing and queried for where data should be placed (table: Canada_Creek_AWS).
- Reader interpreted this correctly, and began building the table.
- Once completed, reader switched to iterating through CSV in parallel, checking and confirming data type, and building each row into a list.
- After each row is written to list, connection to database is opened, list is inserted, committed, and the connection is closed
- With each connection close, row number is printed to terminal for test purposes, confirming it was properly inputted
- Audit of final row printed matched maximum rows of data in csv file
- Process repeated with 2 more months on 5min data

Confirming the database integrity

- Access database
- Confirm table schema [/dt]
- Confirm that before Reader writes its first csv to the database there is no table of the used name (Canada_Creek_AWS)
- Confirm after Reader writes a csv, the table is created in the schema, and after subsequent writes to the same table, duplicates are not created

Querying the database

- Access the database
- Query the database []
- Confirm query provides correct values

- Query database for input into CSV
- Confirm all values are correctly provided

Results of Dev. Testing:

All TFCs passed in preliminary testing, can move to user testing. Alpha declared, pending review from Gwenn Flowers and completion of user testing. Kanban on Github updated to reflect this. (https://github.com/nathanields/Prototype_Glacial_Database/projects/1)

User testing plan

Erik has expressed a need for large amounts of data from the Canada Creek and North Glacier weather stations. Current plan is to load up all data from both of those sites into the database to confirm integrity of CSV reading system for AWS data. Erik will then attempt to query the data he needs for his work from this database. The resulting CSV of all concatenated data will be audited by myself to make sure he is not receiving any incorrect values and once audited can be used for his work.