Parameter Refactoring

Summary

The Geodashboard relies heavily on the measurements taken at a location. The current architecture does not assign unique ID's to measurements taken, so we use conventions *to identify which values in a datapoint are actual measurements that the client wants visible. For example, inside a datapoint you can save anything. It could be this:* ***"temperature": 98.6*** *but it could also be this:* ***"Temp F": "98.6"*** *(a different ID and a string instead of a float). Or it could even be this:* ***"temperatures": ["oh look, I'm an array now!", 98.6, 100.5].***

This worked in the beginning, but clients are starting to ask things like:

- *Can I assign a QAQC value to a measurement (quality assurance, quality control)?*
- *Who modified this QAQC value?*
- *When was it modified?*

Here is how we have done that up until now:

- *"temperature": 98*
- *"temperature-qc": "A"*

Then on the client we use Regex to find a QC value for each parameter. A request to the API for datapoints with parameter "temperature" doesn't return the QAQC code since it thinks "temperature-qc" is a different parameter. It isn't very robust, and the more things we want to add to a parameter, the messier this will get.

We are proposing the nesting of parameters into an array or an object so that more information can be stored within each measurement.

Current Format

- properties
  - temperature: 98
  - wind-speed: 15
  - temperature-qc: "P"
  - wind-speed-qc: "A"

Proposed Format

- properties
  - parameters (ARRAY of OBJECTS)
    - 0
      - id: "temperature" – this should be the exact format as it is found in the raw file that was parsed
      - value: 98
      - qaqc: "P"
      - qaqc-description: "Preliminary"
    - 1
      - id: "wind-speed"
      - value: 15.01
      - qaqc: "P"
      - qaqc-description: "Preliminary"
    - 2
      - id: "wind-speed"
      - value: 15
      - qaqc: "A"
      - qaqc-description: "Approved"
      - modified: 2016-05-09 12:15:00
      - modified-by
        - id: ted-kratchmer
        - first-name: ted
        - last-name: kratchmer
        - email: ted@ted.com

Notes:

- We discussed adding a "published" attribute
- We discussed adding "modified" states and the user that did it

- We discussed making **datapoints.properties.parameters** an object instead of an array. As an array, you'll have to query and find all items where "id" equals your parameter ID. As an object, you could access it directly like **datapoints.properties.parameters.temperature**. There are pros and cons to both of these, but ultimately what we want out of either structure is to be able to add multiple "events" for each measurement. For example, the raw data said "temperature" was 98 degrees. That was the created event. Then Ted came in and modified the value to 97.8 degrees and changed the QAQC code to "Calibrated". This is a modified event and it should do a couple things:
    - It should have a mechanism in place to change the old measurement from **"published": true** to **"published": false**. The new measurement should then be **"published": true.** What I'm typing out here isn't the full solution, we will need to think through all of this, but we wanted to capture the notion that you could have multiple *events* for a parameter's measurement inside a datapoint without all of them being visible in Geodashboard and without all of them being downloaded by default.
    - The modification event should also log who modified it, and all the fields that were modified. It should be thorough enough that we could build an interface that allows us to "undo" all changes made going back to the ingestion of the raw data, and so that you would have an audit log.
    - We can easily add an update function in postgres to add a "modified" timestamp, but we will need to figure out how to add the authenticated user's information to the event.

Transition Plan

- We will continue parsing into the old format until everything is ready.
- Write a script that converts the old format into the new format and make the database dumps available
- Update the datapoints endpoint in Clowder
    - datapoints?attributes= - this currently gets contents of **datapoints.properties** by key. It might be helpful to keep this in case you want to query non-parameters in a datapoint
    - datapoints?parameters= - we might consider adding "parameters". This might only look inside **datapoints.properties.parameters**
    - if we are using a "published" flag in addition to QAQC values in a parameter, we should filter datapoints by those with the "published" flag set to true.
    - we might also consider allowing a logged in user pass in **?published=false** to get unpublished data.
    - If we are using a published flag, when saving a parameter to a datapoint, the API should check to make sure that no other parameter with that ID has a published flag of true, otherwise we might show approved and raw datapoints accidentally
    - update the **sensors**/update endpoint to create an index of the unique id values in **datapoint.properties.parameters** instead of the unique keys in **datapoint.properties**
    - update the **streams**/update endpoint to create an index of the unique id values in **datapoint.properties.parameters** instead of the unique keys in **datapoint.properties**
- Update the Geodashboard Code so that it looks in **datapoints.properties.parameters.**

Team

- Jong Lee
- Luigi Marini
- Brock Angelo
- Eugene Roeder
- Marcus Slavenas
- Indira Gutierrez Polo