Task

Design, write in Python 3 and run a prototype application that processes measurement data from the MQTT message broker. The broker is powered asynchronously.

Description

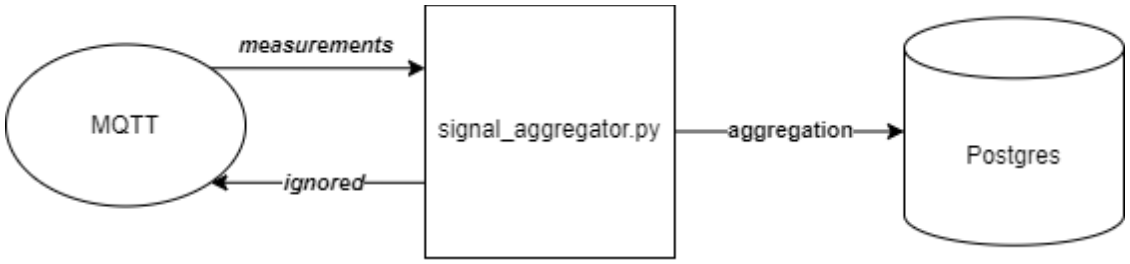
- The application reads messages from the subject: ***measurements****. These are* measurement data in json format:

```
{
 time: "2022-01-01T00:00:00Z",
 value: 2001.01,
 unit: "V"
}
```

- Aggregate this data at intervals of T seconds and save the following result in PostgreSQL after obtaining N measurements for the aggregation period:

period_start	period_end	N	min	Max	median	average

- The parameters T and N should be given as arguments to start the application.
- Measurements that appear after saving the aggregation should be sent back unchanged on MQTT ***ignored***



Welcome

- Justification of the project decisions taken omitted from the description
- Description of the limitations of the proposed solution
- Automatic test
- A solution that runs as docker.

Solution Description/Plan:

Publish_data:

- Connect to MQTT broker, for publishing data
- Data is generated randomly with given JSON format with exact time stamp, continuously every (0.3-1) s

Process_data:

- Subscribe to MQTT broker with subject name: *"measurements/#"*
- def *on_conenct():* start getting messages(data)
- Aggregate the data.
- Connect with local PostgreSQL database
- Insert to database (**store_data.py**)

** Each next run will append the database table "measurement" with new data

Aggregation of data:

- User will give the time interval "T" (in second) and number of measurement "N" they want to take.
- Then program aggregate the data(messages) collected in time of "T" and repeated for "N".
- Each "N" times aggregated data converted into table of "N" rows, then store it in database.

- Example: T = 5, N=3

  Suppose program will receive 10 messages within T time.

  > program aggregated 10 messages
  > repeated for N times

  >Total messages received during process 30

  > number of rows = 3, since N is 3

Welcome:

Justification of the project decisions taken omitted from the description:

- Measurements that appear after saving the aggregation should be sent back unchanged on MQTT ***ignored***
    - Here, the Idea of sent back unchanged measurements is not clear by saying “sent back on MQTT ignored” since the program can subscribe to the topic name on MQTT broker and receive the message, or it can unsubscribe after **ignored** to stop receiving messages.
    - The program is *exit* after the saving the result.

Description of the limitations of the proposed solution:

- Proposed solution is not tested with, large amount of time interval "T", which can be resulted into memory management.
- Need to change Code manually for database connection
    - (host= “localhost”) to run in local machine
    - (host="host.docker.internal") for docker solution
        - Can be solve with proper docker-compose scripts with describing network connection
- publish data, process data can be done with class-based programming (OOP)

Automatic test

- The automatic test is omitted due to lack of functionally. May be, project was not architecture with clear functional way, where different components can be tested.

A solution that runs as docker

- Docker solution, describe in README.md file as installation