neOCampus DATA: user-guide

Dr Thiebolt François, thiebolt@irit.fr

This document is the DATA user guide for neOCampus end-users.

Modifications table

Date	Note
jun.20	added per sensor downsampling
may.20	Initial release

Abstract

This guide will explain to you the various ways to gain access to the neOCampus DATA asset.

Table of contents

Overview

It's been a long time since neOCampus started to collect data from our sensors and actuators spread over our campus. The very first data was retrieved from U4 building, room 302 in 2017, june! It was our first attempt to collect these data in our mongoDB database.

Quite recently, we setup a new data storage architecture based on the **data lake** approach: the storage of our data is splitted across 3 zones as shown in fig.1:

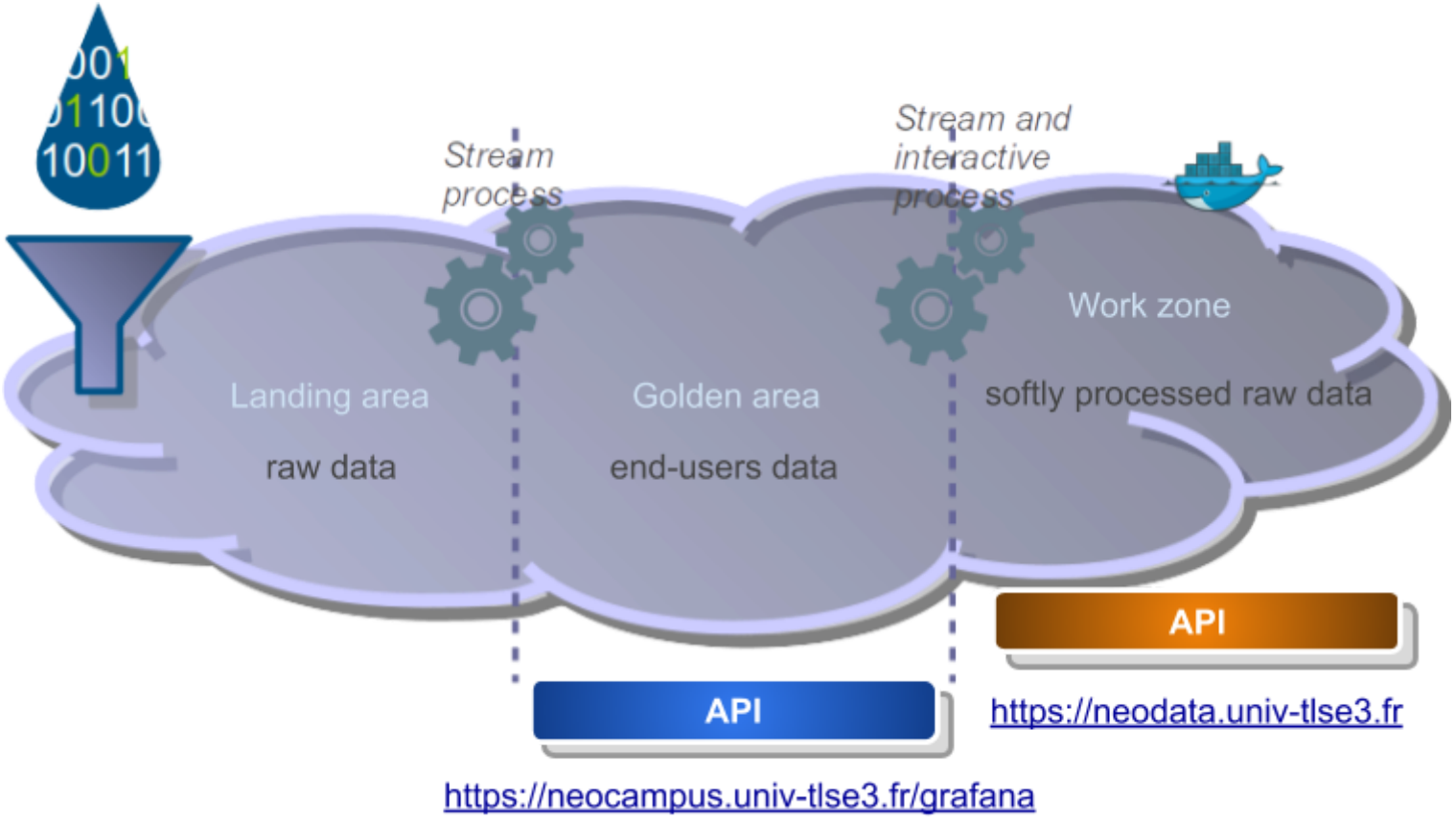


Fig. 1 - the neOCampus 'data lake' like.

Here is a short description of these 3 zones from our data lake:

Zone	Description
landing area	raw data coming from sensors; we just add a timestamp.
work zone	some soft processing has been applied to raw data producing 'more understandable' data (mostly data have been set with tags describing their location)
golden area	the real 'end-user' area with fully processed data showing a per location (i.e site, building, room) view featuring weekly, monthly and yearly synthesized data.

neOCampus data lake | first release

More than one year ago, we were facing issues accessing data from our mongoDB database: while the data was time indexed, it used to take, sometime, up to ten minutes to retrieve data for a specific room or building … those days are gone :)

While talking with some of our colleagues and our internship applicant[1] a few months ago (feb.20), the idea of a multi-database storage area with stream processing in-between started to emerge. We tried to take into account the fact that different kinds of people will express different kinds of access to these data. Our end-users (university services, building managers etc) would like to have a synthetic view of these data while our PhD (kind of intermediate users) on their side would need to launch some specific processing on (almost) raw data.

While not really being a data lake as all of the storage and processing are self-contained in one server, we expressed a three tiered data storage architecture with stream processing in-between. One day, we'll probably switch to a real data lake, so it ought to be just a matter of switching from one local data source to a remote one … at least we hope!

In the remaining part of this short overview, we'll give you some details about the different storage areas, we'll also present you the actual data sources, then next chapter will delve within the real data access procedures.

landing area

At this time of writing (may.20), we're using a **mongoDB** database as our **landing area**. These raw data (~100million entries ---may.20) are indexed according to the timestamp of the measure. Here are the ways to access these data:

- interactive GUI → https://neocampus.univ-tlse3.fr/mongo
- client library to mongoDB
  (e.g Python3 pymongo https://api.mongodb.com/python/current/python3.html)

We must confess that the mongo-express UI is not really relevant for legacy data (i.e before apr.20 ---see fig. 2) access purposes. One way to circumvent this issue is to allow you to undertake your data processing in an offline manner: working directly on a mongoDB dump. If you're willing to go this way, please send an email to the neOCampus management team https://neocampus.org

work zone

Maybe the most interesting area for PhD and other people working on data to extract knowledge and other stuff. A few weeks ago we setup a new time series database: **InfluxDB 2.0**[2] This new release combines a data access API, (light) visualization tools, interactive query editor and many bindings for a broad range of client libraries. By the beginning of may, we imported all of the data available in mongoDB ⇒ you now have access to the broad range of data starting from 2017 :)

[1] Pr André Péninou, Pr Olivier Teste, Dr Jacques Thomazeau, M. Vincent Nam-Dang, Dr Hamdi BenHamou
[2] https://www.influxdata.com/products/influxdb-overview/influxdb-2-0/

There exists several ways to gain access to these data:

- interactive GUI → https://neodata.univ-tlse3.fr
- client library to influxDB 2.0
  (e.g Python3 https://github.com/influxdata/influxdb-client-python)
- HTTP Restful API (better is to go through a client library),,
- offline processing through InfluxDB dumps.

golden area

This area will get dedicated to end-users that need a synthetic overview. You 'll have a dashboard allowing you to select your data according to your needs (e.g location, building). Then more detailed graphs about rooms for example will feature: a one week view (5mn resolution) and up to one year view (1 day resolution with min, max, mean). It will be something pretty close to what Domoticz offers.

Going one step further, in addition to the per sensor aggregated data, we may propose you a per room and building synthesized data that will leverage your needs for high-level end-users UI.

Ok, here are the following ways to access these data:

- Dashboard UI → https://neocampus.univ-tlse3.fr/grafana
- client library (may be a mix of InfluxDB and mongoDB)

Synthesis

Zone		
landing area (raw data)	before apr.20	[mongoDB] neOCampus DB: 'measure' collection 'failedData' collection
	after apr.20	[mongoDB] neocampus_datalake DB: 'sensors' collection
work zone (softly processed data)	from may.17	[influxDB] 'sensors' bucket (*legacy data were imported)* 'metrics' bucket (*MQTT metrics*)
golden area (synthesized data)	up to 1 year ago	[influxDB] *for all sensors:* 'sensors_hires' bucket (*5mn resolution across 7days)* 'sensors_lowres' bucket (*1day resolution across 1year)*

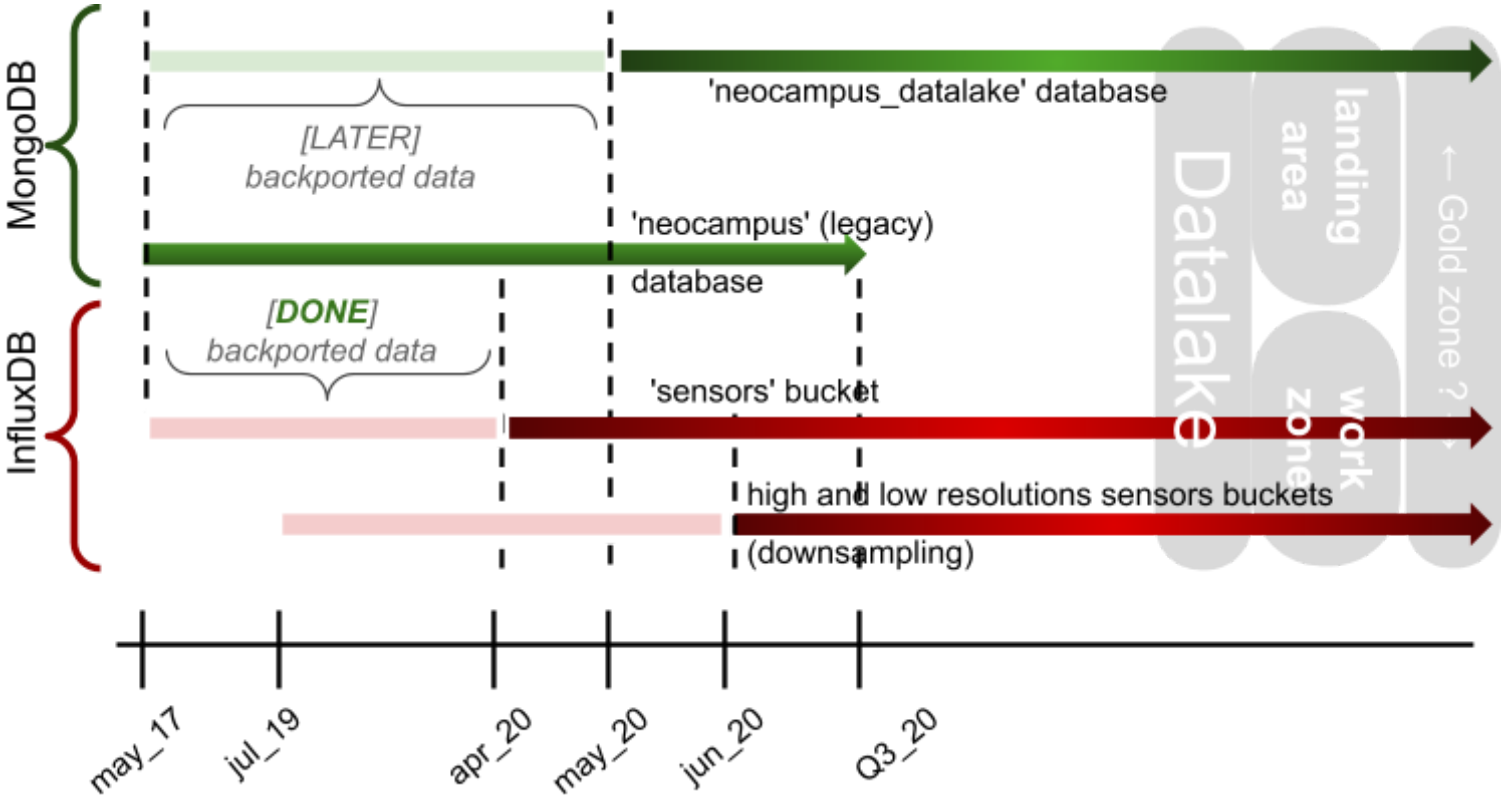


Fig. 2 - neOCampus data timeline.

As stated in fig. 2, a bit later we'll re-import all of the 'legacy' data to the neOCampus data lake 'landing area'.

neOCampus data sources

Below is a description of the various data sources brought to you.

Date	Description
	Apiary data
	LoRaWAN end-devices data
	SGE data
apr.20	[UT3] BU sciences and IRIT2 buildings
apr.20	Toulouse metropole weather station
jun.17	[UT3] U4 building

Note: greyed items means that data are not yet available.

Data and API access

Of course, all thoses access require credentials, either as login / password (e.g mongoDB) or tokens (e.g influxDB). Our students may just drop me an email. Pay attention to the fact that most of these API are only reachable from UT3 networks.

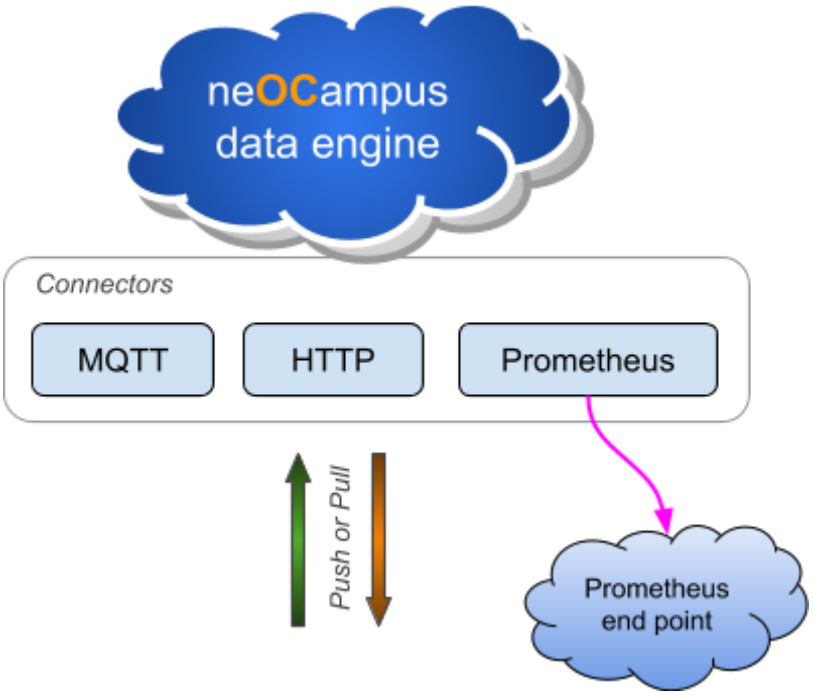
adding new data sources ?

We encourage you to propose new data sources that can be taken into account and that will benefit everyone.

We propose several connectors for both live and offline data:

- HTTP(s) in both PUSH or PULL modes,
- Prometheus protocol to collect data from end-points.

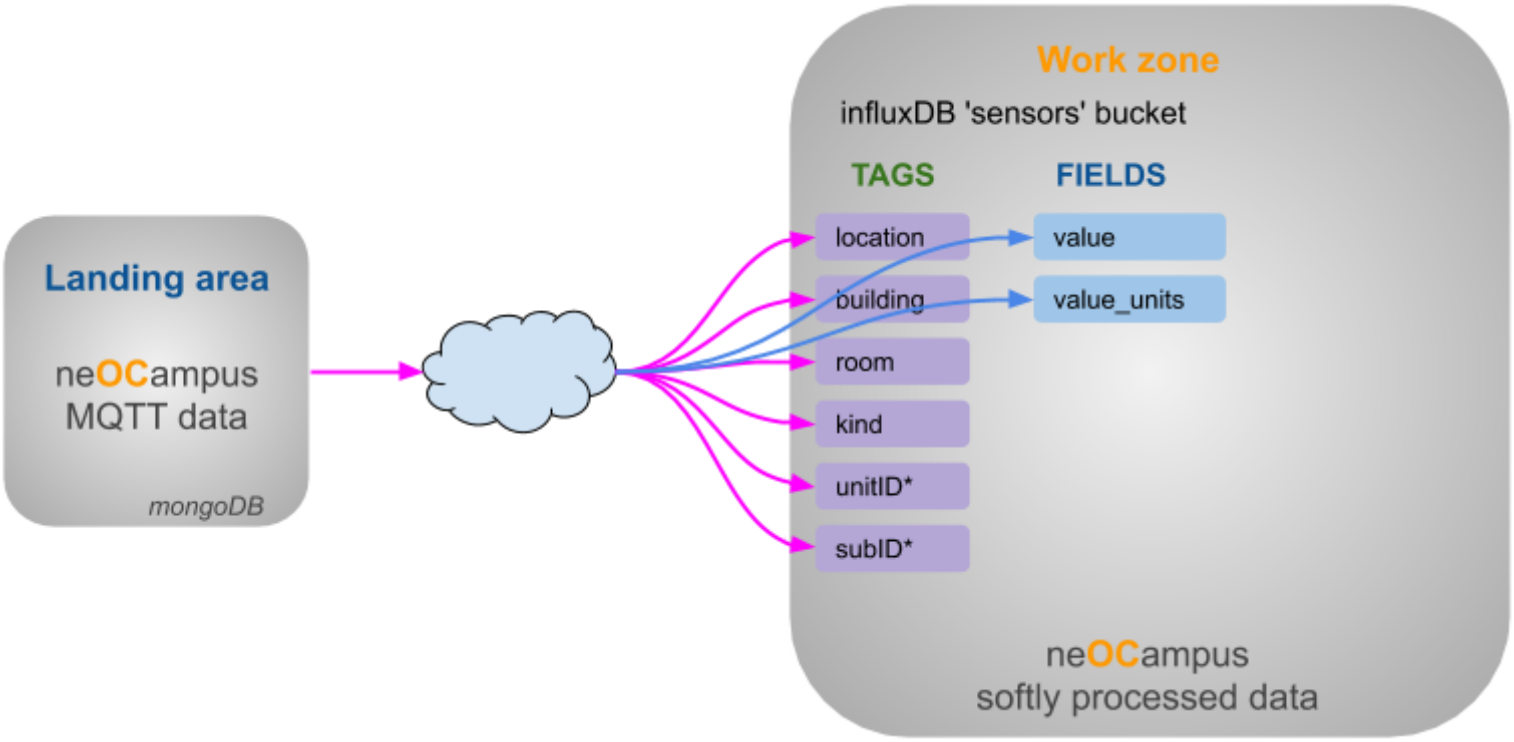
In case your app. does not behave as a prometheus end-point, we can add a prometheus gateway that will make your data easily reachable.



[work zone] DATA access

We're using influxDB to store our sensors' data. Before going on with data access, we need to specify their format.

As stated earlier, neOCampus MQTT data get stored in the **'landing area'**. These data whose format is specified in [1] get slightly processed to finally arriving in the '**work zone**'



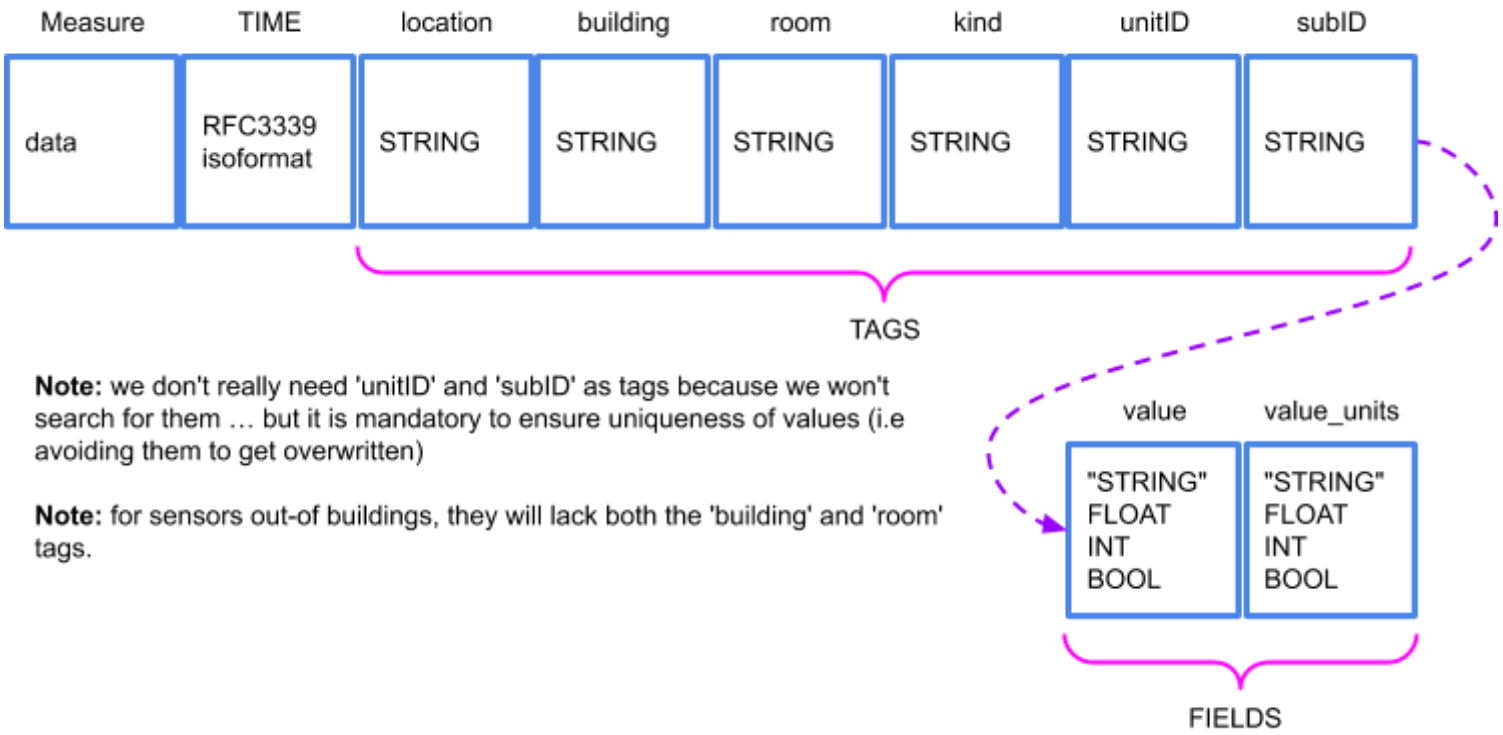
'sensors' bucket structure

Hold all of the collected data from sensors / actuators.
This bucket only features one measurement:

- data

'data' measurement

TAGS	@data measures [sensors bucket]
location	e.g ut3, ensimag, carcassonne, ...
building	e.g u3, u4, ...
room	e.g 300, 301, 302, ...
kind	e.g temperature, wind, humidity, energy, ...
unitID	*as in payload*
subID	*as in payload*
FLAGS	**@data measures [sensors bucket]**
value	*as in payload*
value_unit	*as in payload*



Note: we don't really need 'unitID' and 'subID' as tags because we won't search for them … but it is mandatory to ensure uniqueness of values (i.e avoiding them to get overwritten)

Note: for sensors out-of buildings, they will lack both the 'building' and 'room' tags.

Note: the various types (i.e kinds tag) are described in [1].

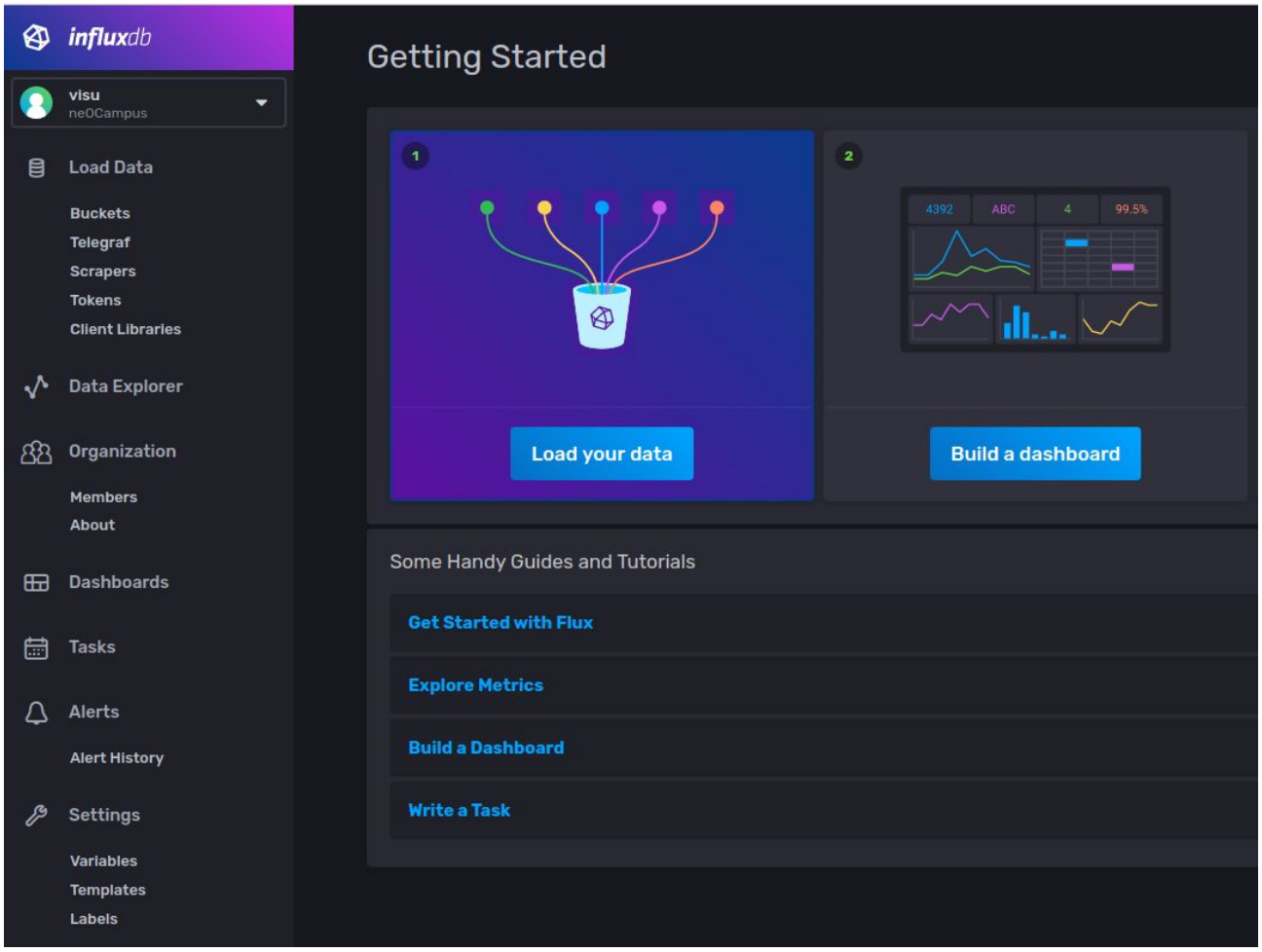
[get data] UI & CSV

InfluxDB natively proposes an UI which allows you to both see and export data in a CSV format.

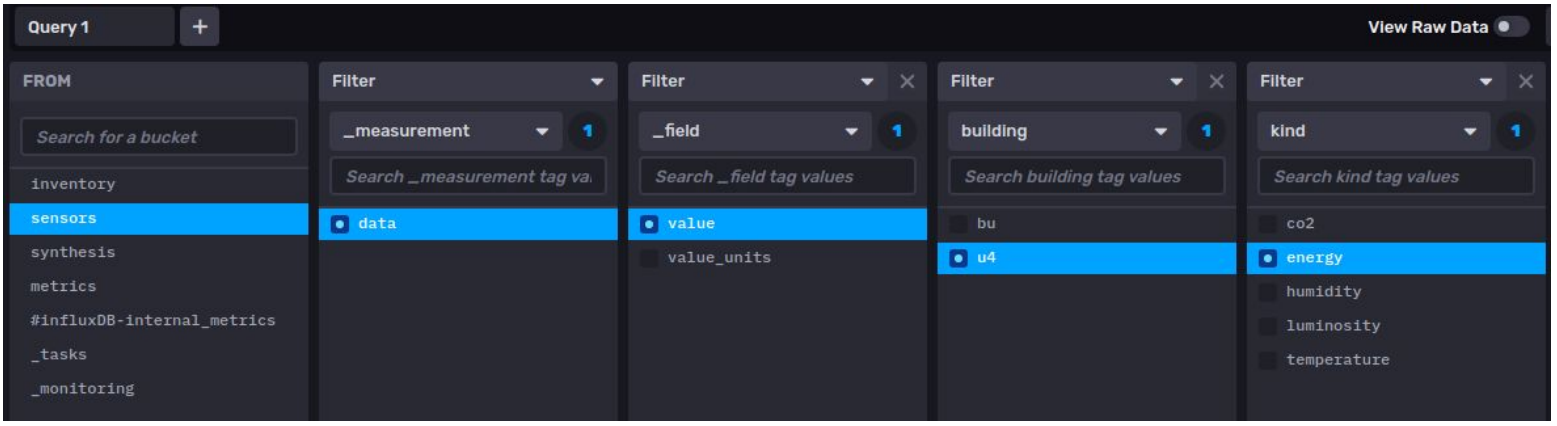
https://neodata.univ-tlse3.fr

login: visu
passwd: <ask us>

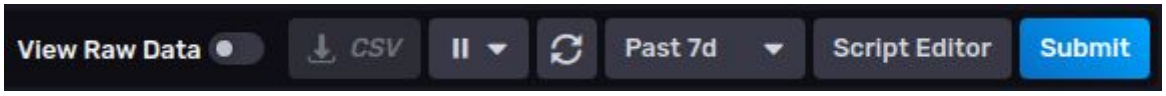
… you then get connected to influxDB



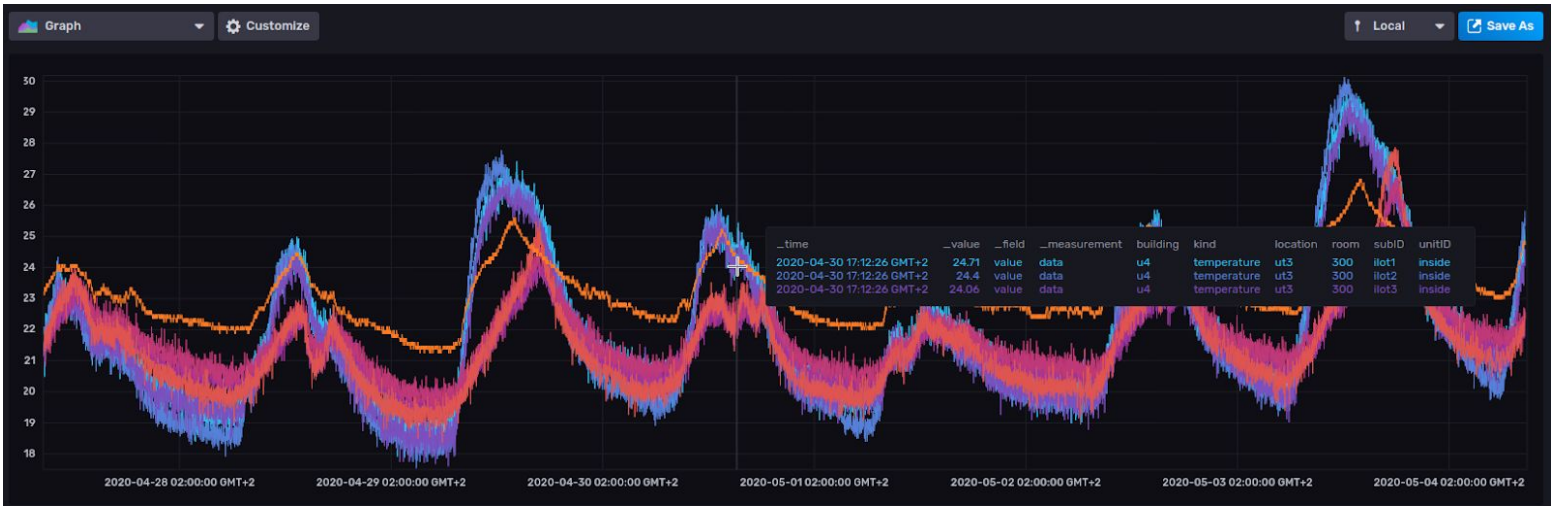
On the left side panel, select 'Data explorer'
→ It will show you several data sources (buckets).
→ select 'sensors' bucket along with 'data' measurement (the time serie itself)
→ you can then start to explore the various buildings, rooms, kinds, …



you can then choose a period of time (e.g last 7d), then click 'submit'



… tada !



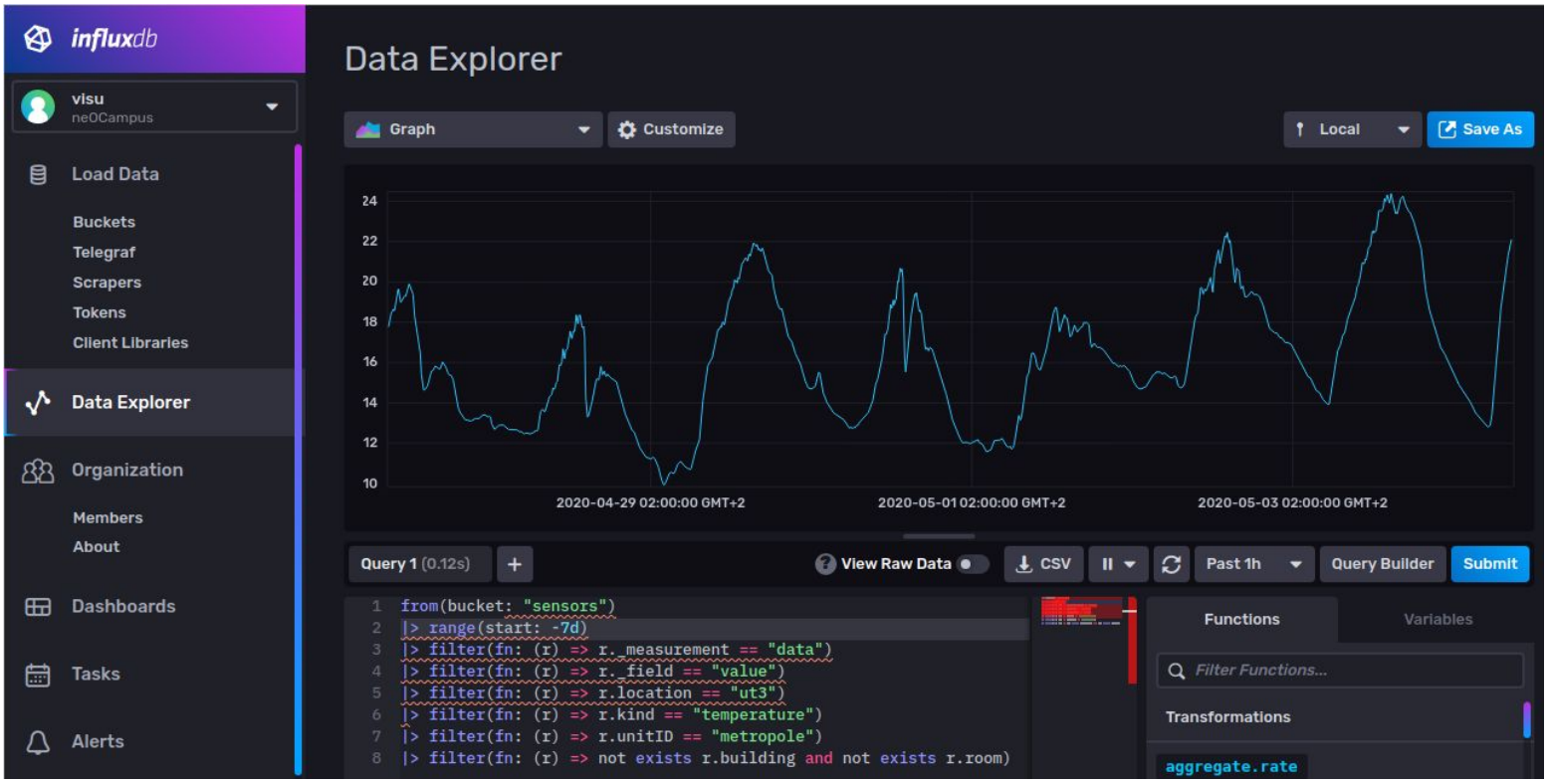
You can also choose to select 'RAW DATA' (tables) … and export as CSV for further analysis.

how to see 'outside' sensors ?

As described in [1] along within the dataCOllector document, outside sensors does not feature any building nor room tags → query editor

```
from(bucket: "sensors")
|> range(start: -24h)
|> filter(fn: (r) => r._measurement == "data")
|> filter(fn: (r) => r._field == "value")
|> filter(fn: (r) => r.location == "ut3")
|> filter(fn: (r) => r.kind == "temperature")
|> filter(fn: (r) => r.unitID == "metropole")
```

```
|> filter(fn: (r) => not exists r.building and not exists r.room)
```



Of course you can undertake more complex Flux queries, see languages details at https://v2.docs.influxdata.com/v2.0/reference/flux/

[get data] client library

Probably the simplest way to undertake your data analysis, there exists various client libraries to gain access to the influxDB database.

server: neodata.univ-tlse3.fr
port: 9999
read token: <ask us>

Our docker container 'dataCOllector' features numerous apps written with the Python client https://github.com/influxdata/influxdb-client-python

```
pip3 install influxdb-client
```

[golden area] DATA access

We'll be providing several kinds of data in this area:

- per sensor data **aggregation** for week (high resolution) or up to one year (low resolution),
- [later] per room, building **synthesized** data.

[aggregation] high and low resolution buckets

These buckets are **downsampled** sensors according to the following rules:

- sensors_hires → **5mn** resolution for all sensors across 1 week (min, avg, max),
- sensors_lowres → **1 day** resolution for all sensors across 1 year (min, avg, max)

These buckets only features one measurement:

- data

Notes:

- TAGS are the same as in sensors bucket,
- every 5mn, all buckets may get updated (downsampling propagation)

'data' measurement

TAGS	@data measures [sensors-WMY buckets]
location	e.g ut3, ensimag, carcassonne, ...
building	e.g u3, u4, … **None** (i.e outside)
room	e.g 300, 301, 302, … **None** (i.e outside)
kind	e.g temperature, wind, humidity, energy, ...
unitID	*as in payload*
subID	*as in payload*
FLAGS	**@data measures [sensors-WMY buckets]**
value_avg	*payload average*
value_min	*payload min*
value_max	*payload max*
value_unit	*payload units*
_avg_count	*counter for avg computation* *(only in low resolution aggregation buckets)*

Measure	TIME	location	building	room	kind	unitID	subID
data	RFC3339 isoformat	STRING	STRING	STRING	STRING	STRING	STRING

TAGS

Note: TAGS will be the same as in 'sensors' bucket

value_avg	value_min	value_max	value_units	_avg_count
"STRING" FLOAT INT BOOL	"STRING" FLOAT INT BOOL	"STRING" FLOAT INT BOOL	"STRING" FLOAT INT BOOL	"STRING" FLOAT INT BOOL

FIELDS

high resolution

5mn synthesis

_time = 13:05:00

data related from 13:00:01 to 13:05:00

low resolution

1 day synthesis

_time = 2019-08-21T00:00:00

data related to 2019-08-21 from 00:00:00 to 23:59:59

[synthesized data] end-users rooms and building data

TO BE CONTINUED

[view data] Grafana

TO BE CONTINUED

[landing area] DATA access

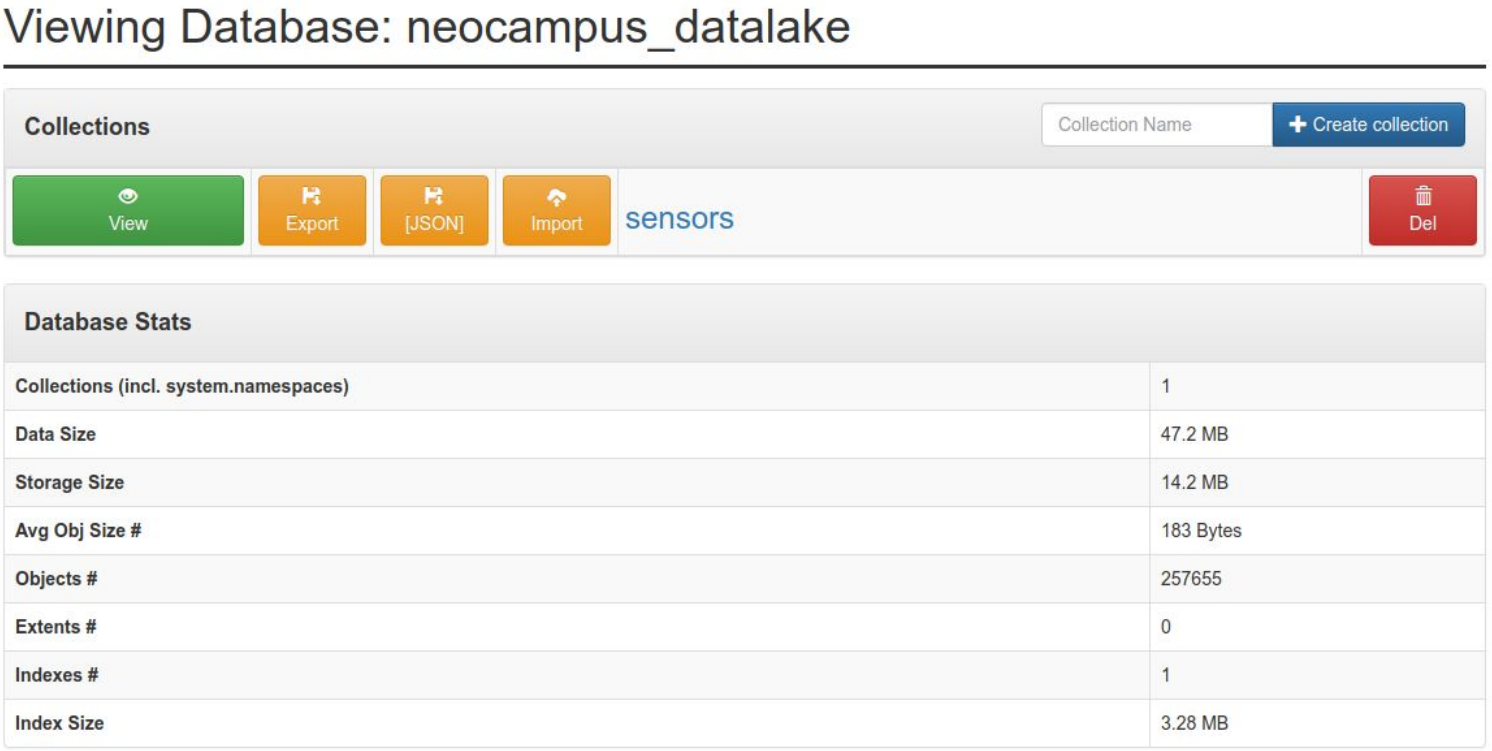
TO BE CONTINUED

[get data] UI & CSV

mongoDB can be accessed through the mongo-express UI front-end.

server: https://neocampus.univ-tlse3.fr/mongo/

login: reader
passwd: <ask us>



[get data] client library

Probably the simplest way to undertake your data analysis, there exists various client libraries to gain access to the influxDB database.

server: neodata.univ-tlse3.fr
port: 27012
login / passwd: <ask us>

Our docker container 'dataCOllector' features apps written with the Python client
https://api.mongodb.com/python/current/tutorial.html

```
pip3 install pymongo
```

Annexe - A | References

[1] neOCampus end-devices API and MQTT rules
This document describes the way end-devices are managed through the sensOCampus application. It also describes the various topics you may encounter in the 'landing area'. More important, all kinds of CLASS topics (i.e temperature, pressure, wind, energy …) are fully specified in this document.
https://neocampus.univ-tlse3.fr/_media/sensocampus_end-devices_api.pdf