XENONnT Offline Monitoring

23 june 2023

Quentin Pellegrini, LPNHE

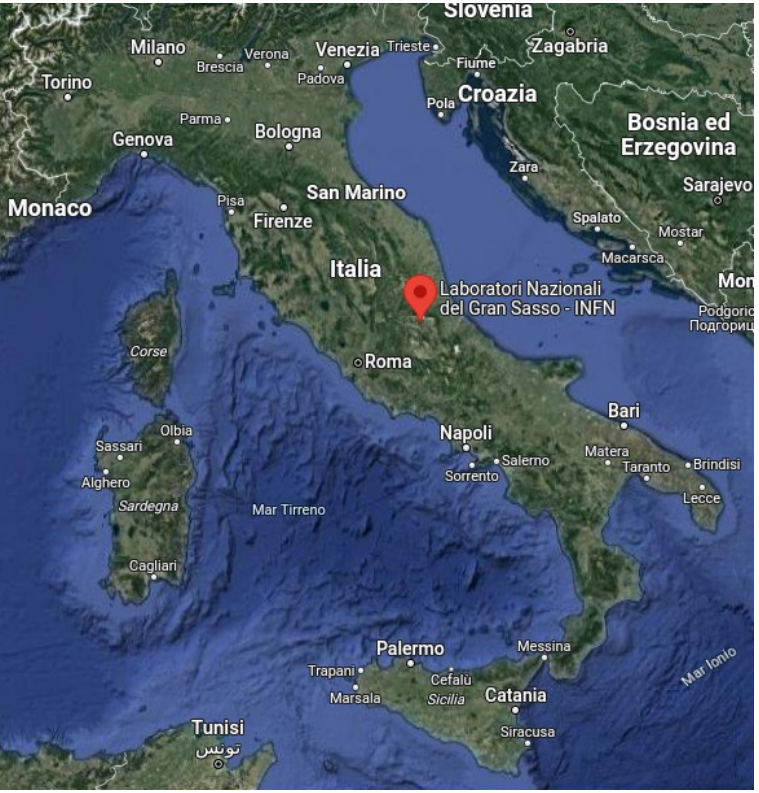
Summary

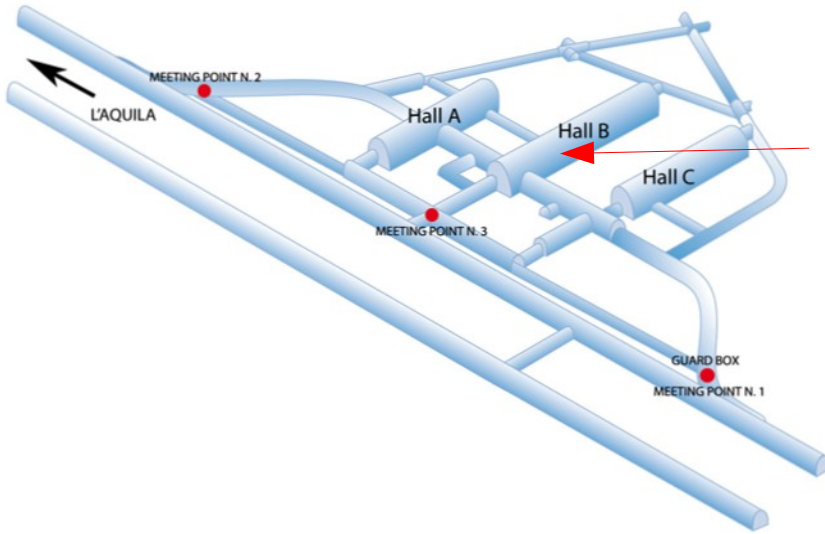
- Data management on XENONnT

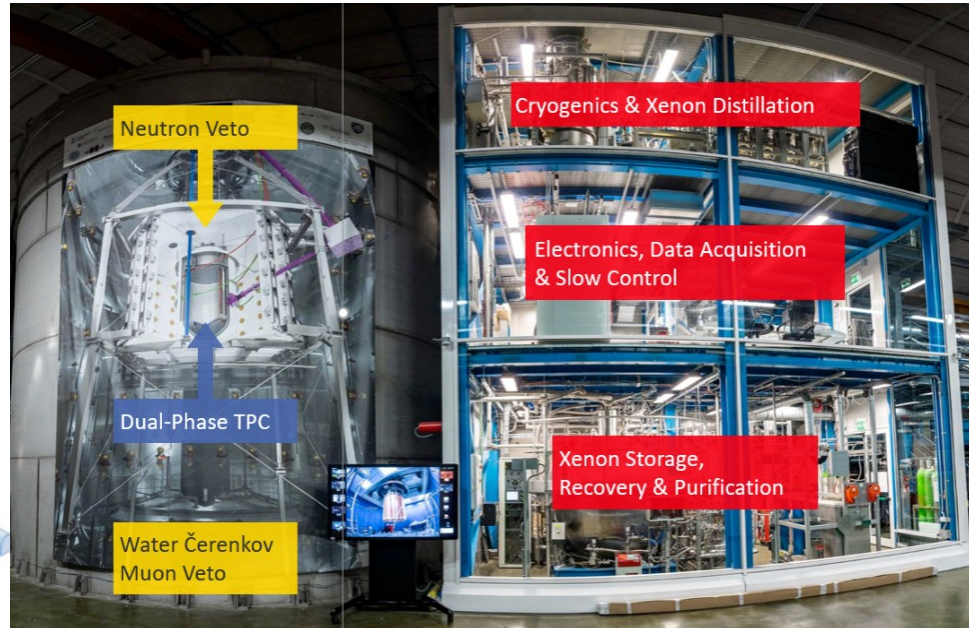
- XENONnT Offline Monitoring (XOM)

XENONnT

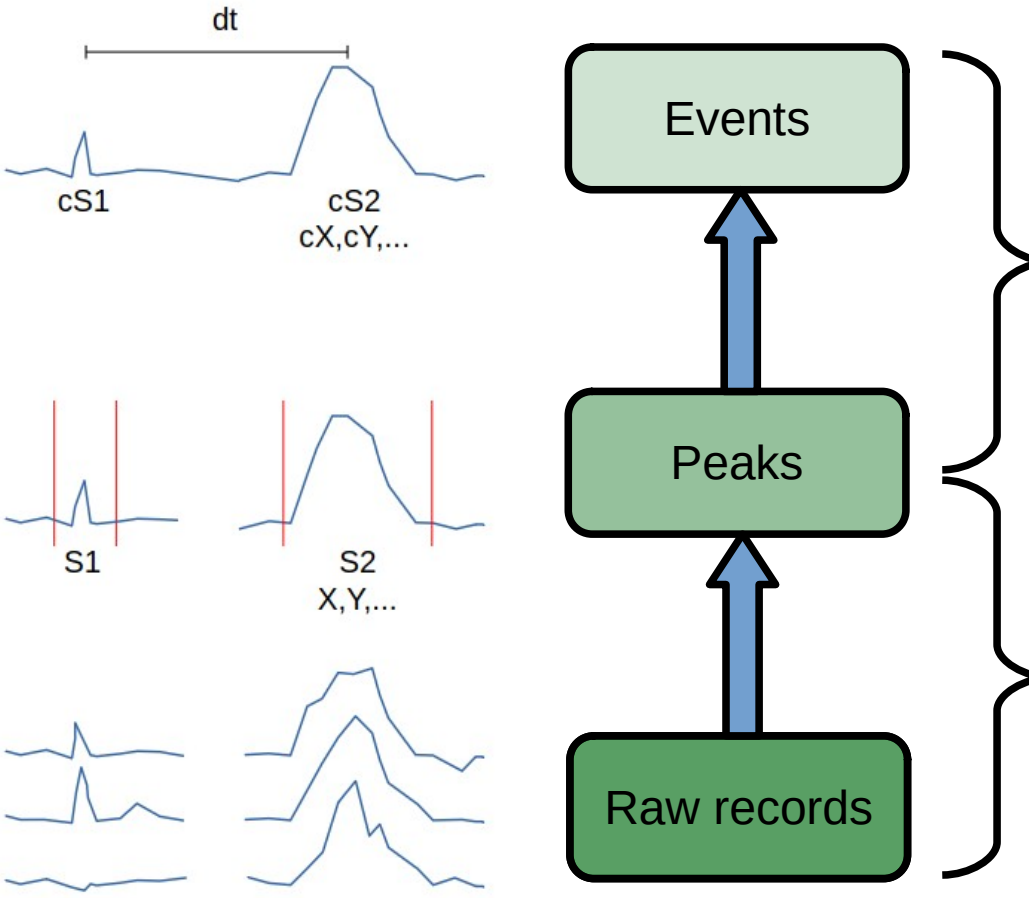
XENON







Data structure



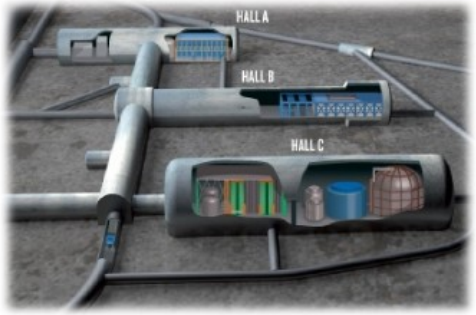
data processing performed by **strax software**

XENONnT monitoring analyses browse most of datatype set
→ Source of complexity

Data Flow

Run DB : Data base for runs (name, tags, location, availability...)

52896	tpc,muon_veto,neutron_veto	background_linked	2023-06-17 21:34 05:00:03	_sr1_preliminary phase1
52895	tpc,muon_veto,neutron_veto	background_linked	2023-06-17 16:33 05:00:03	_sr1_preliminary phase1
52894	tpc,muon_veto,neutron_veto	background_linked	2023-06-17 11:32 05:00:03	_sr1_preliminary phase1



DAQ Event Builders
LNGS underground

66 TB

Datamanager
LNGS on surface

62 TB

US and EU GRID sites
Raw data

~8 PB

UChicago
USD
CC-IN2P3
NIKHEF
SURFSARA
CNAF

UChicago RCC "Midway" GRID
Processed data

200 TB ↔ Jupyter

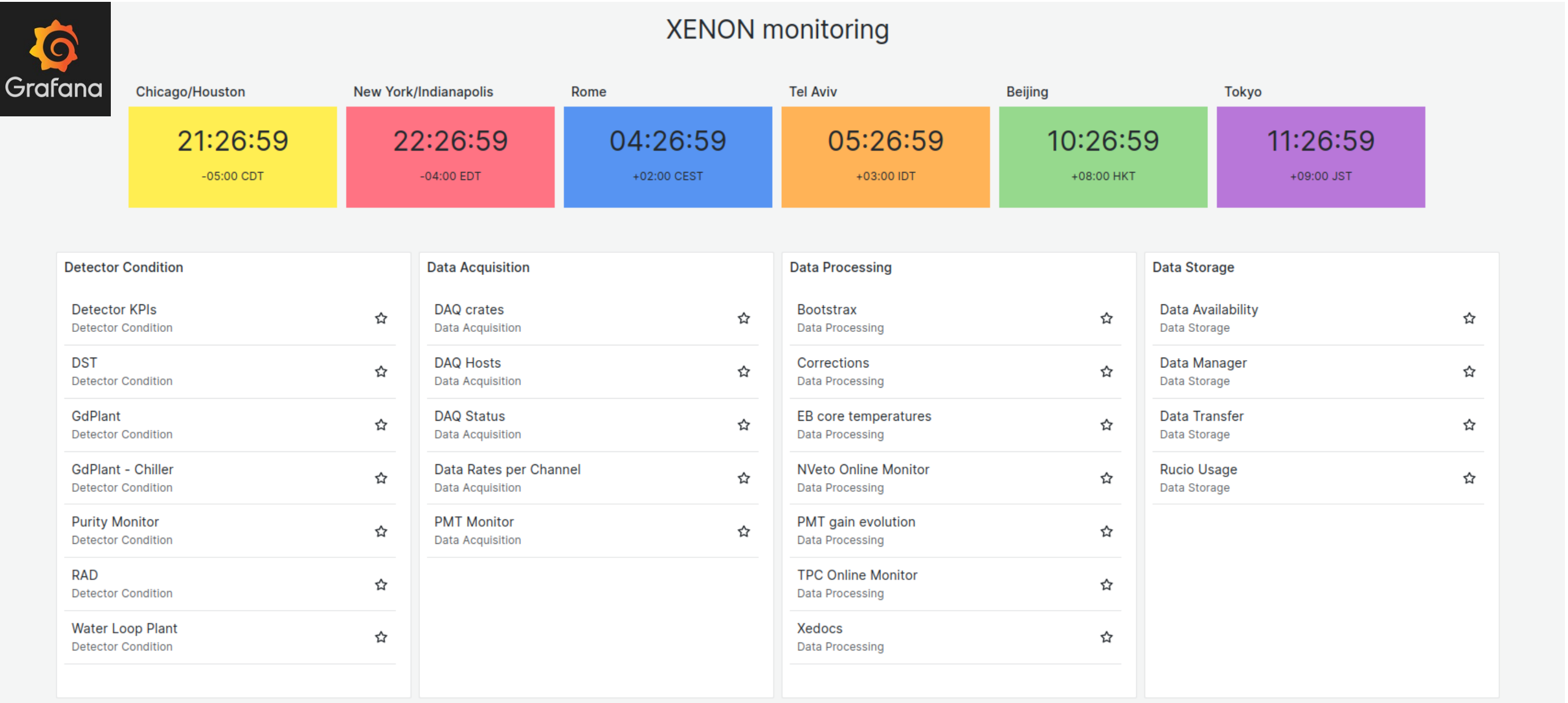
Data flow handled by aDMIX (*github.com/XENONnT/admix*) and Rucio (*rucio.cern.ch*)

Average rate (including calibrations) : 1 PB / year

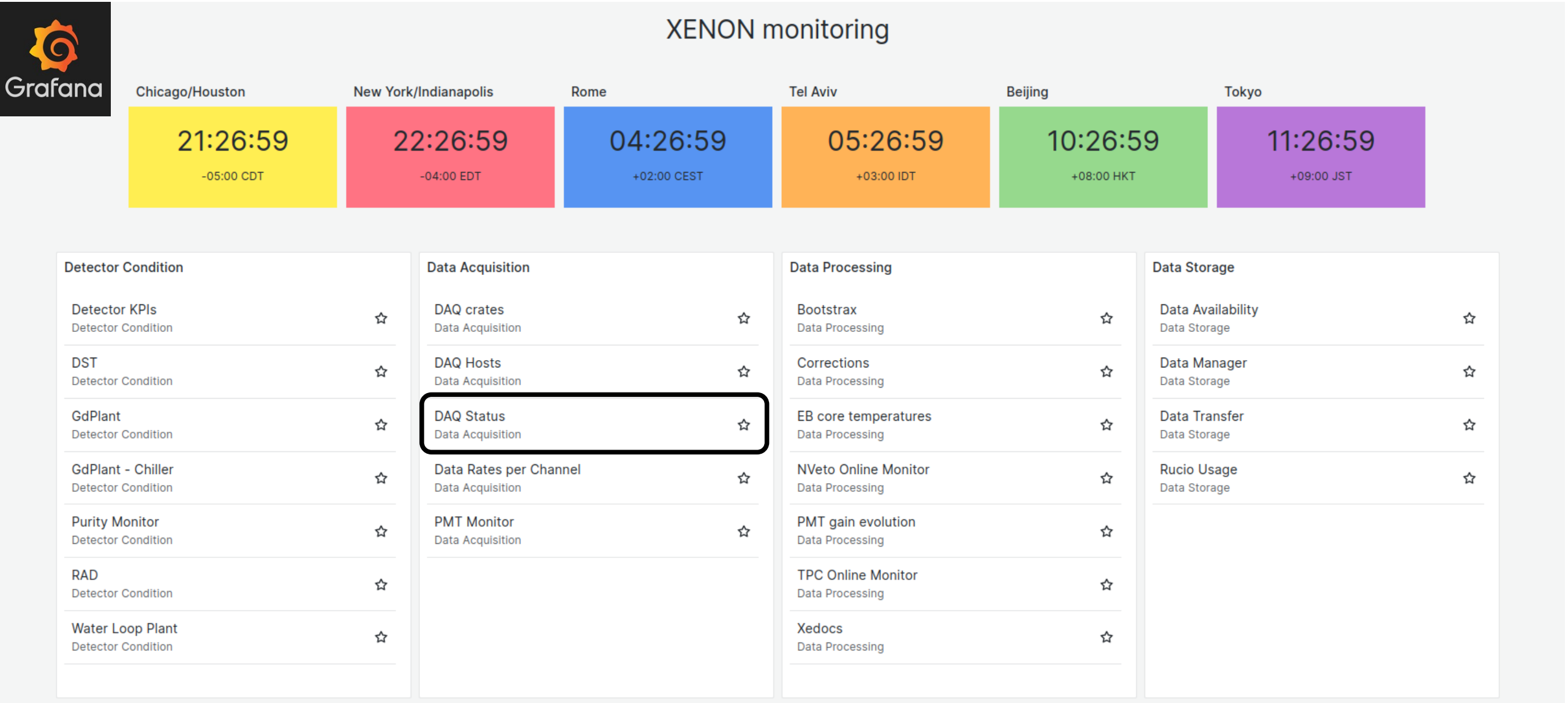
Redundancy x2 for raw data:
- disks+tapes (current science run)
- tapes+tapes (previous runs)

Monitoring needs data from multiples sites
→ Data availability complexity

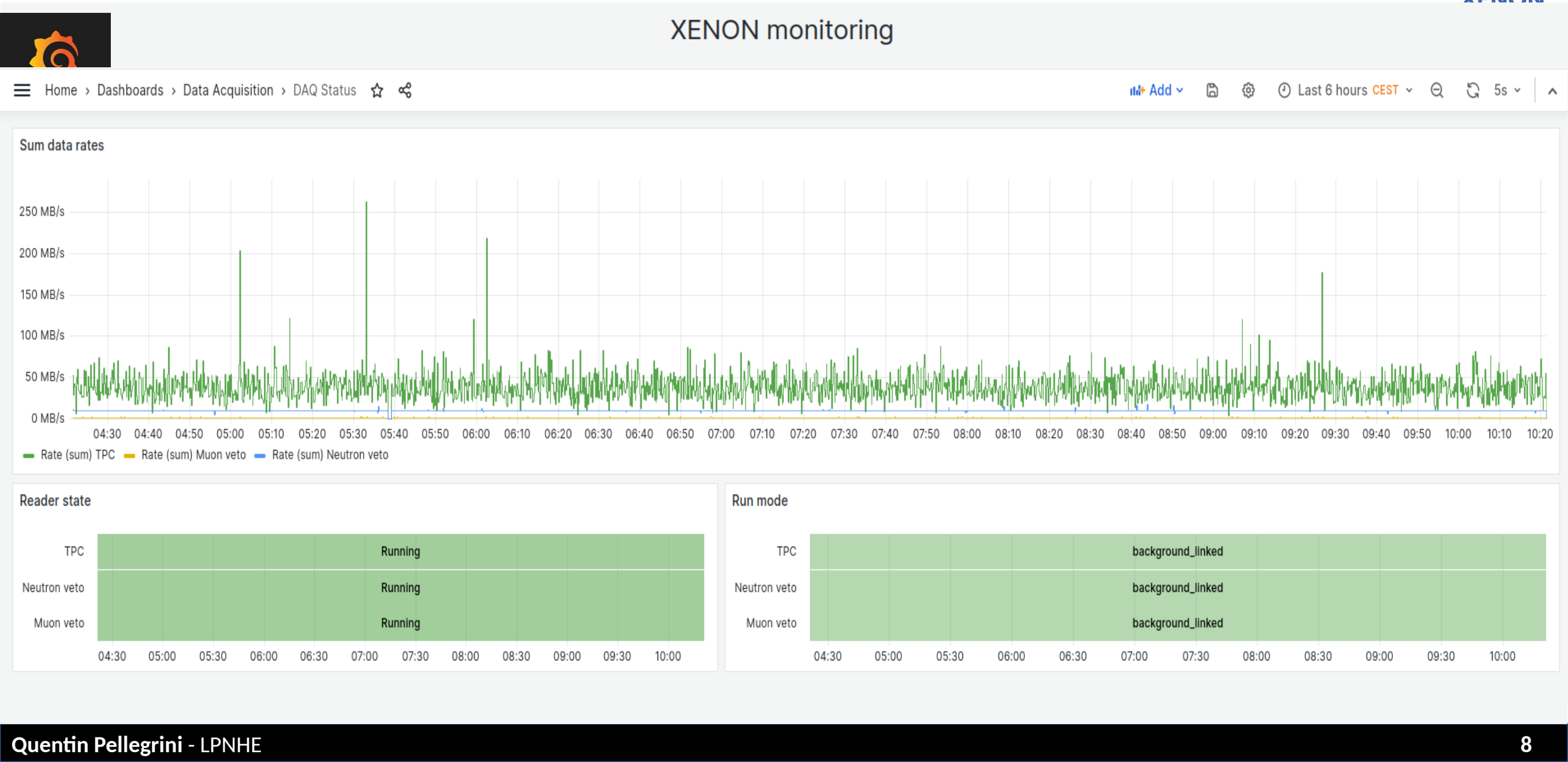
XENONnT Monitoring Status



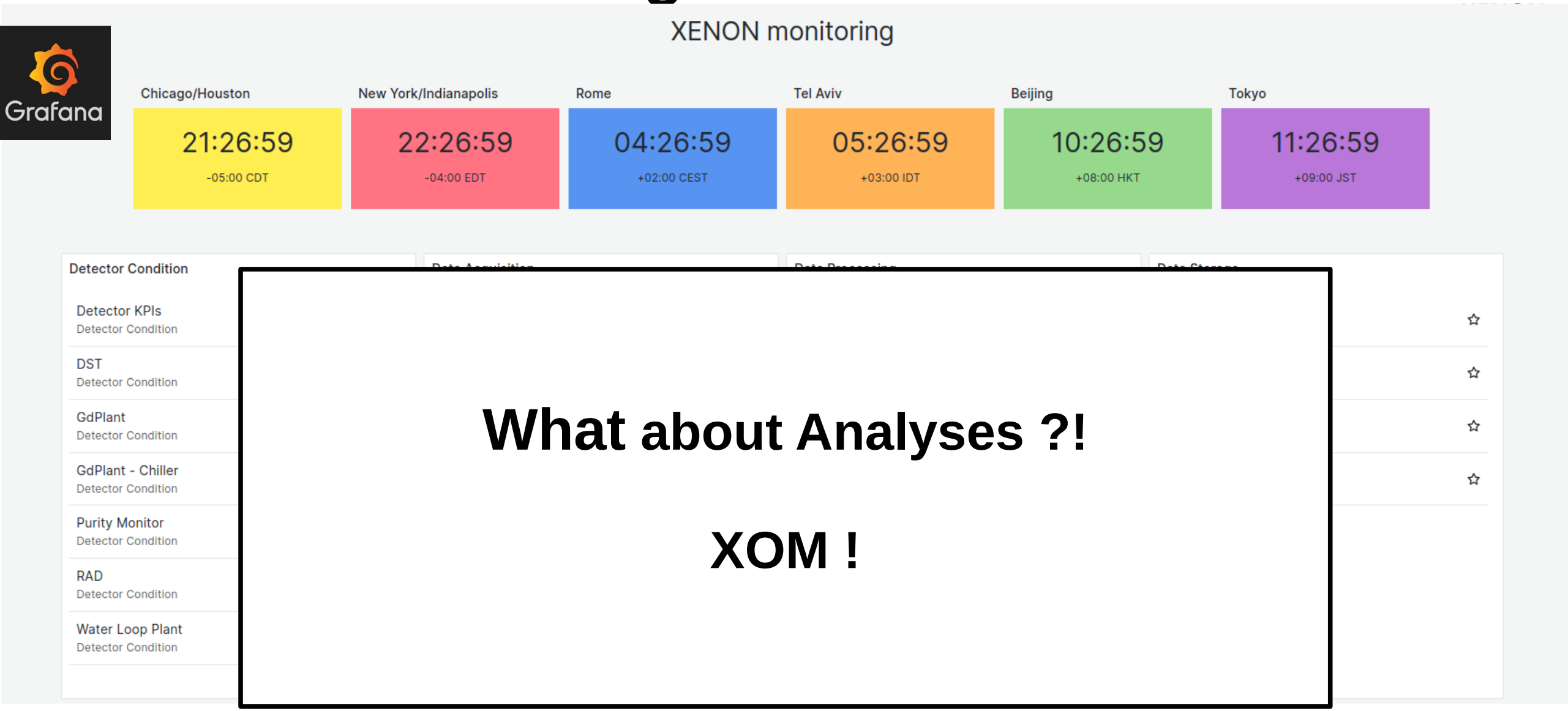
XENONnT Monitoring Status



XENONnT Monitoring Status



XENONnT Monitoring Status



What about Analyses ?!

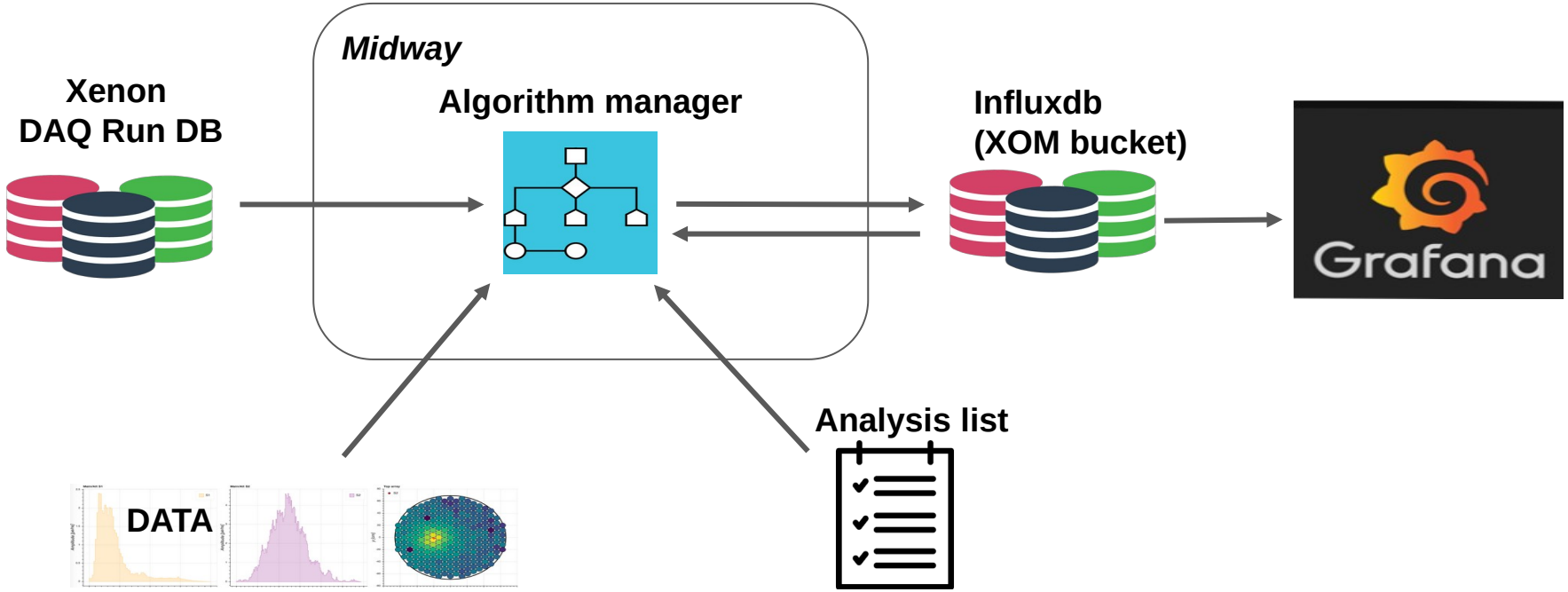
XOM !

XENONnT Offline Monitoring (XOM)

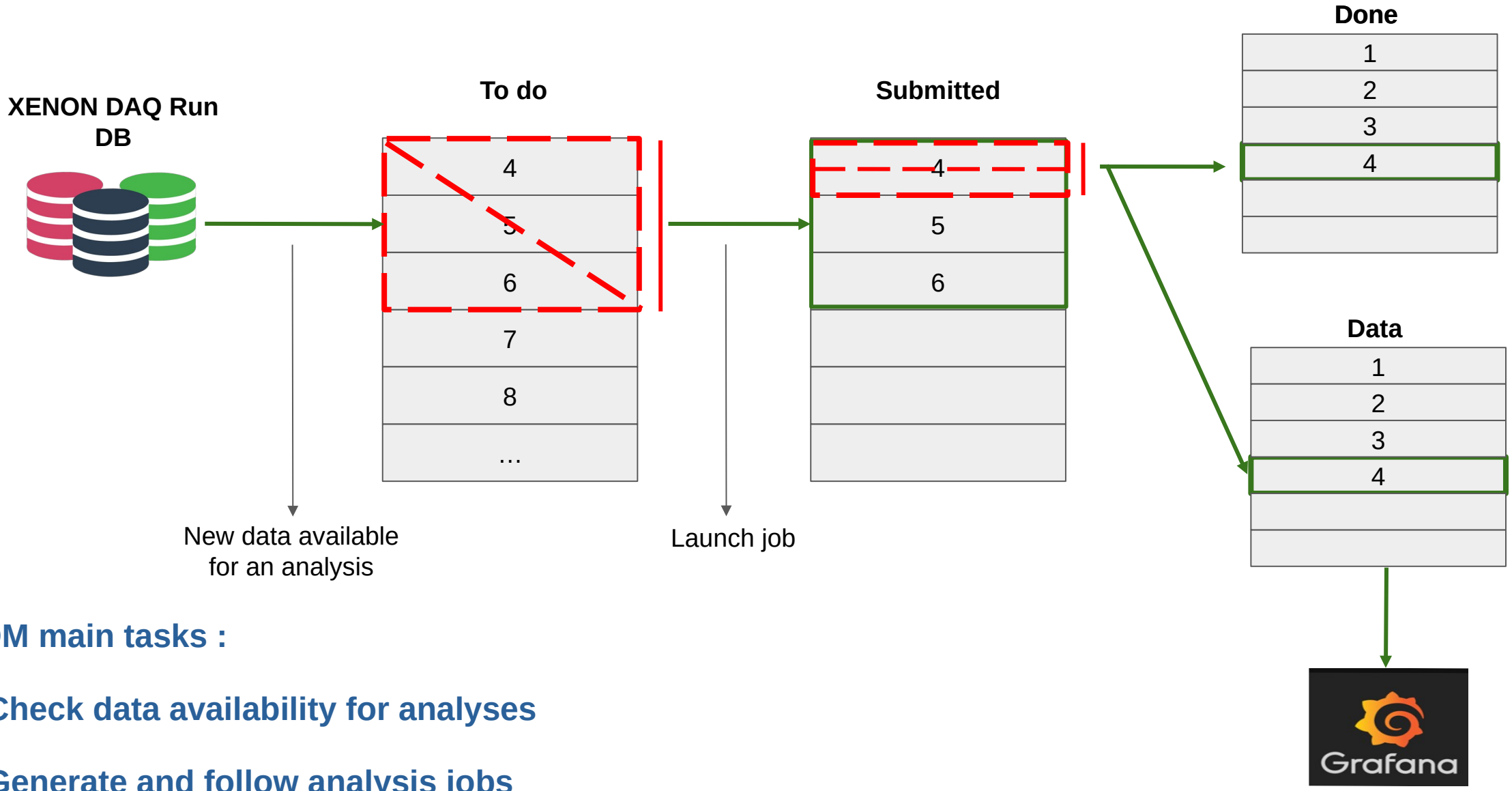
Recent Tool developed within the xenon team (LPNHE) by Romain Gaior and me.

Allows automotically running analyses as new data appaears (independantly of complexity, data type...)

Analysis results are displayed in XENON Grafana



Xenon Offline Monitoring (XOM)

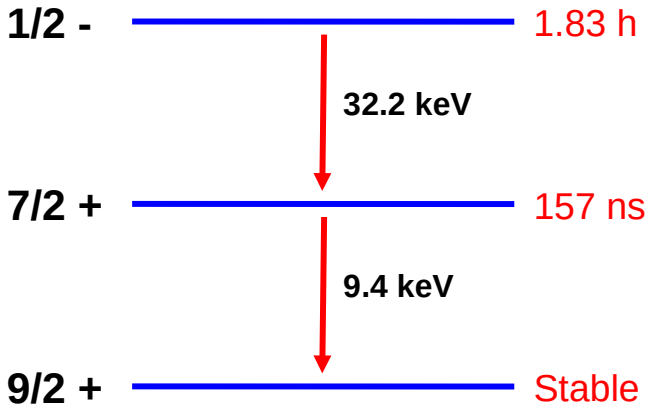


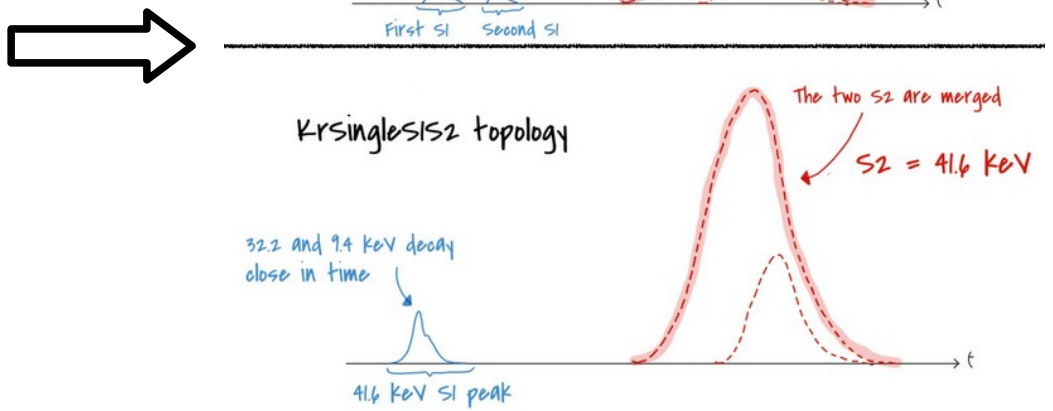
XOM main tasks :

- **Check data availability for analyses**
- **Generate and follow analysis jobs**

An example !

Kr83m bimonthly calibration





Parameters

$$LY = \frac{S1}{E}$$

$$CY = \frac{S2}{E}$$

$$S2(E) = S2(0)e^{-\frac{\Delta t}{\tau_e}}$$

LY (9.4, 32.2)
CY (41.6)
Electron lifetime

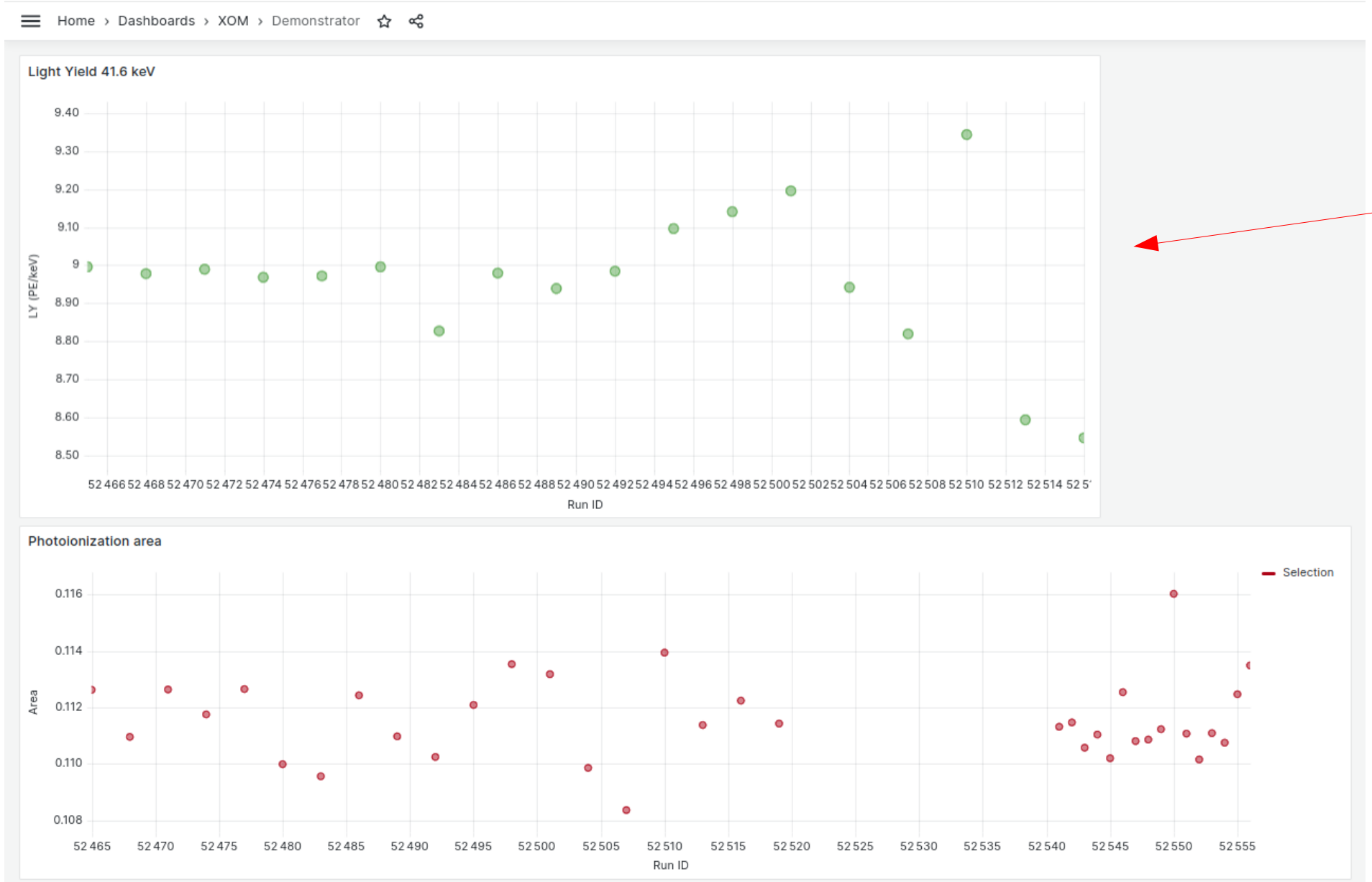
LY (41.6)
CY (41.6)
Electron lifetime

Krypton Calibration Script :

- **Data selection for a Krypton run thanks to Maxime Pierre**
- **Calcul of the mean via simple mean or gaussian fit**
- **Compute LY, CY**

XENON

XOM and Grafana



Krypton LY

Conclusion

- XOM is currently running four analyses (Krypton LY, Event number, Scada, Photoionization area)
- New analyses are coming soon (Muon and neutron veto…)
- We will improves XOM robustness as new alayses are added

- XOM will centralize and automate analysis for monitoring
- It is an excently tool to test data availability and data quality
- XOM responds to a strong need for XENON collaboration

- Data base management and Graphana
- Data availability, analysis context and diversity
- Visibility, responsibilities and interactions in XENON

Thank you for your attention

23 june 2023

Pellegrini Quentin, LPNHE

Xenon Offline Monitoring (XOM)

proc_compare.py
Load analysis_list

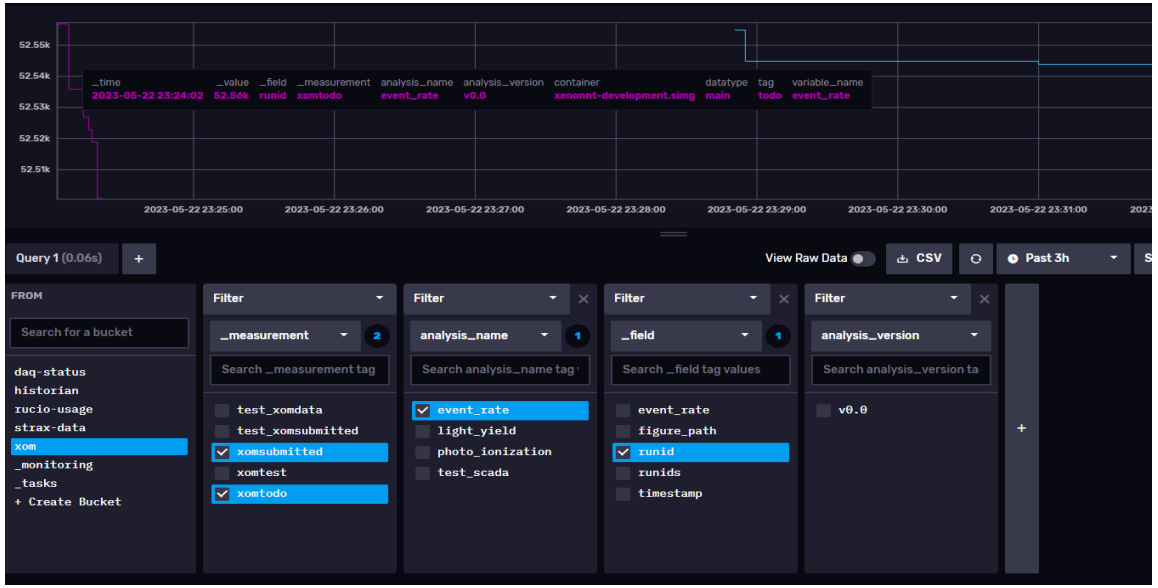
If new_run in rundb:
    For *analysis* in *analysis_list*:
        If *new_run* is good for *analysis:*
            Add entry in TODO db

proc_runner.py
Load analysis_list

For *analysis* in *analysis_list*:
    Check data availability (st.select_run)
    Store the list of run available

For entry in TODO db:
    If running number of jobs < limit:
    If data is available:
    Submit the job to do
    Add entry in the SUBMITTED db
    Delete entry in the TODO db

For entry in SUBMITTED db:
    Check status of the job
    If finished:
    Add entry in DONE db
    Delete entry in the SUBMITTED db




```
import xomlib

main(run_id):
#load the data
st = cutax.xenonnt_online(_rucio_local_path='/project/lgrandi/rucio', include_rucio_local = True)
st.storage += [strax.DataDirectory('/project2/lgrandi/xenonnt/processed', provide_run_metadata=True)]

# do whatever interesting
# data processing here

# create a xom object with the analysis result
xomresult = xomlib.Xomresult(analysis_name="photo_ionization",
                             analysis_version = "v0.0",
                             variable_name='area',
                             variable_value=area,
                             runid=int(run_id),
                             data= {"area":area, "rate":rate})

# simply adds a message in the log
# save the result in the DB with the correct format
xomresult.xom_message(success=True)
xomresult.save()
```