PIC
port d'informació científica

El Port d'Informació Científica

Services

IFAE Institut de Física d'Altes Energies

GOBIERNO DE ESPAÑA MINISTERIO DE CIENCIA E INNOVACIÓN Ciemat Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

BIST Barcelona Institute of Science and Technology

CERCA Centres de Recerca de Catalunya

UAB Universitat Autònoma de Barcelona

Unió Europea Fons Europeu Next Generation

GOBIERNO DE ESPAÑA

Plan de Recuperación, Transformación y Resiliencia

Next Generation Catalunya

Generalitat de Catalunya Departament de Recerca i Universitats

PIC
port d'informació científica

Outline

- PIC Introduction
- Landing at PIC
- PIC services
  - Batch system: HTCondor
  - Storage
  - Notebooks ecosystem: Jupyter
  - Version Control System: Gitlab
  - Big Data: Cosmohub & Spark

Port d'Informació Científica

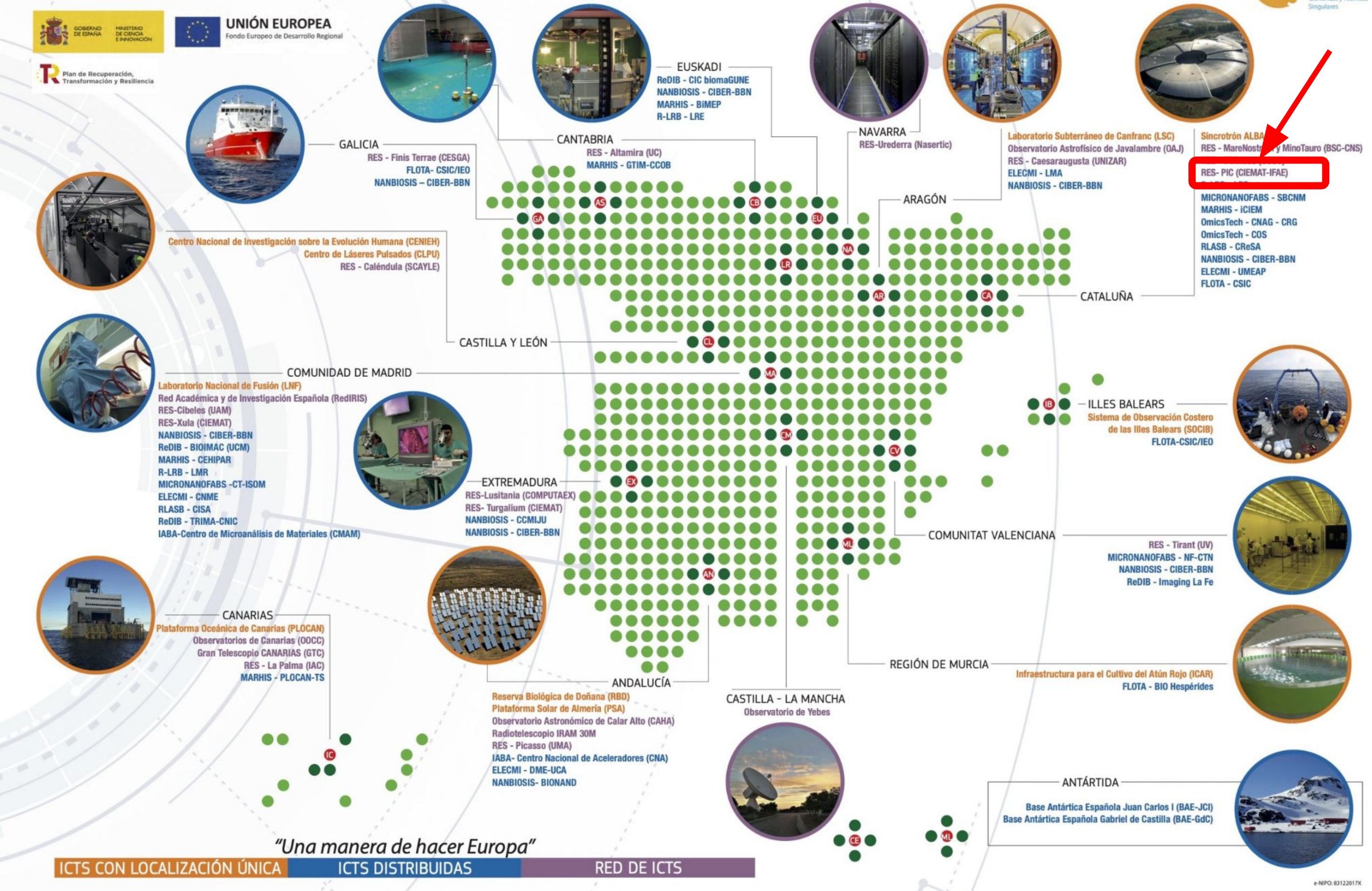
- Founded in 2003: collaboration between IFAE and CIEMAT
- Team of 23 people (50% scientists - 50% engineers)
  - Agile teams that embed in scientific groups to
    - Understand the experiment
    - Follow the evolution of data analysis requirements
    - Develop & prototype tools for data management and analysis
- What we do
  - R&D in methodologies and tools for advanced data analysis.
    - Participate in R&D projects. Software and Computing WPs.
  - Operate services for the preservation, analysis and sharing of data.
    - Run prod. services for experiments: LHC, MAGIC, CTA, PAUs, Euclid, VIRGO/LIGO, DUNE
    - Provide data analysis services for research groups: IFAE, CIEMAT + others

Red Española de Supercomputación

- RES was founded in 2007 as a network of supercomputers in Spain.
    - Competitive calls for CPU resources open to all research groups in Spain.
- In 2020 RES increased its scope to include **data services**
    - PIC joined RES in 2020.
    - Annual calls for data projects published in December.

https://www.res.es/

MAPA DE INFRAESTRUCTURAS CIENTÍFICAS Y TÉCNICAS SINGULARES



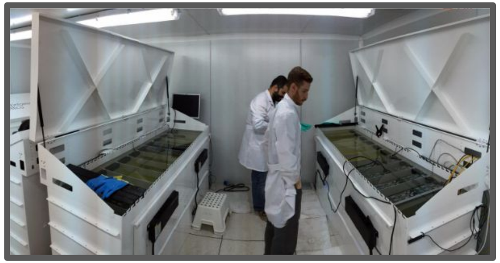
PIC data center

- Connectivity
  - 2x100 Gbps to Academic Network
  - Largest data mover in Spanish academic network: 100 PB in+out per year

- Data processing services
  - Disk - dCache: 20 PB
  - Tape - Enstore: 63 PB
  - Computing - HTCondor: 12000 cores, 18 GPUs
  - Computing - Hadoop: 720 cores, 2.5 PB disk

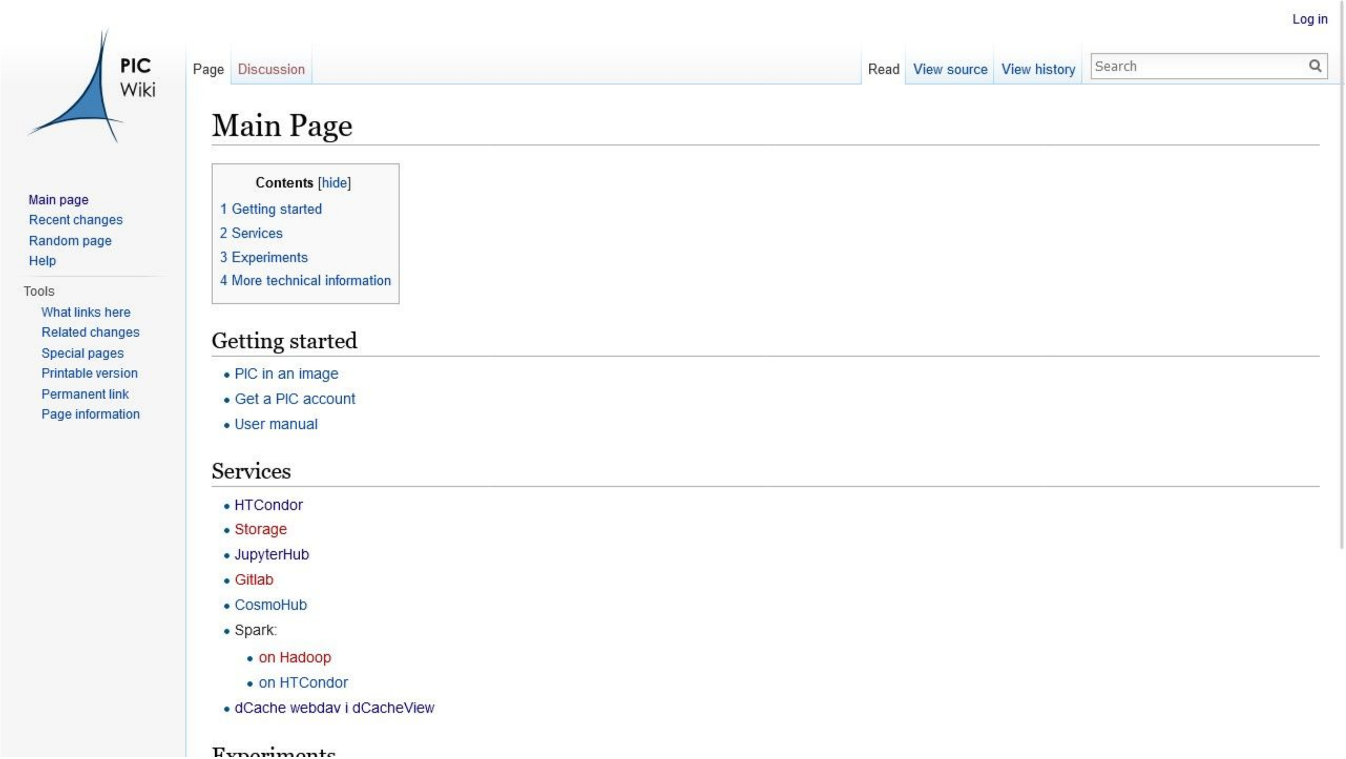
- Facilities, ~120 kW IT
  - ~80 kW in 150 m^2 air-cooled room
    - high efficiency, PUE 1.44
  - ~40 kW in 25 m^2 liquid immersion cooling system
    - very high efficiency, PUE 1.1

IBM TS4500



PIC public wiki

https://pwiki.pic.es



Get a PIC account

PIC Account Request Form

Show Help

Applicant

Salutation: --

First Name *

Last Name *

Work Telephone * +34 612 34 56 78

Work e-mail *

Confirm e-mail *

Sponsor Information

Name of Legal Entity *

Name of Contact Person *

Email of Contact Person *

Relationship with Entity * Employee

Affiliation

Select a main group.

Optionally, select any secondary group that you wnat or you think you should be part of. Your petition will be accepted by the experiment/s contact/s.

ATTENTION: If you can't find a group or you have additional information, specify it in the field below.

Primary Group * Select your main group

Secondary Group(s) *(optional)*

Use CTRL+Click (or CMD+Click in MacOS) to select multiple subgroups

Additional Information

HTCondor: Introduction

- HTCondor is a batch system
  - Software that manages the execution of several other programs (called jobs)
  - It allows the execution of a program without the direct supervision of the user
  - The execution can be in remote machines
- Other popular batch systems: SLURM, Torque, LSF, SGE
- HTCondor is a batch system meant for High-Throughput Computing (HTC) while others are for High-Performance Computing (HPC)
  - HTC favours the execution of a lot of jobs consuming few resources each (CPU, memory, etc.)
  - HPC favours the execution of parallel computing, few jobs consuming a lot of resources (MPI, low-latency interconnection)

PIC port d'informació científica

HTCondor: Introduction

- Batch systems generally rely on queues
  - Users submit jobs to queues that have different limits (maximum time, CPUs per job, etc.)
  - The jobs are executed in the remote WorkerNodes (WNs) according to a priority and the resources available
- HTCondor is different. Your job has requirements and the WNs have resources, if they match, your job will run
- HTCondor uses the fair-share concept
  - You or your group have a quota, a percentage of resources you can use. It is allowed to exceed that quota but your priority will be reduced the next time

HTCondor: Introduction

- HTCondor scales to thousands of jobs and resources
  - There are ~160 WNs, ~12000 cpus and 18 GPUs in our cluster
- We manage more than 20 projects in the same cluster. From the large grid experiments (LHC, CTA, etc.) to the local ones
- HTCondor User Guide:
  - https://pwiki.pic.es/index.php?title=HTCondor_User_Guide

HTCondor: Access the cluster

UI (User Interface)

- To obtain access to PIC HTCondor cluster you can ssh to ui.pic.es
- UIs have access to shared data and allow you to submit and check your job status. They are NOT for interactive execution

```
$ ssh user@ui.pic.es
user@ui.pic.es's password:
Last login: Wed Oct  4 10:40:06 2023

********************************************************************************
*                                                                              *
*                  The "ui.pic.es" Public Login Unix Service                   *
*                             CentOS 7.8.2003 x86_64                           *
*                                                                              *
* Our Login Service grants you access to PIC datacenter and its batch          *
* system (HTCondor) on batch and interactive mode (condor_submit -i).          *
* We recommend consulting our HTCondor User Guide:                             *
*                                                                              *
* https://pwiki.pic.es/index.php?title=HTCondor_User_Guide                     *
*                                                                              *
* In case of problems, please contact the support:                             *
*           e-mail      user.support@pic.es                                    *
*                                                                              *
*                                             PIC WebSite: http://www.pic.es/  *
*                                                                              *
********************************************************************************
[user@ui02 ~]$
```

HTCondor: Quick Start

Submit file: communicate everything of your job to HTCondor (test.sub)

```
$ cat test.sh
#!/bin/bash

/bin/stress $@
```

```
$ condor_submit test.sub
Submitting job(s).
1 job(s) submitted to cluster 7952359.
```

```
$ cat test.sub
executable = test.sh
arguments = -c 1 -t 60
output = condor.out
error = condor.err
log = condor.log

queue
```

```
$ condor_rm 7952359
All jobs in cluster 7952359 have been marked for removal
```

condor_submit to submit your job, **condor_q** to monitor, **condor_rm** to kill

```
$ condor_q 7952359


-- Schedd: submit01.pic.es : <193.109.174.82:9618?... @ 10/26/23 15:43:44
OWNER   BATCH_NAME     SUBMITTED   DONE   RUN    IDLE  TOTAL JOB_IDS
cacosta ID: 7952363  10/26 15:43      _      _      1      1 7952359.0

Total for query: 1 jobs; 0 completed, 0 removed, 1 idle, 0 running, 0 held, 0 suspended
Total for all users: 1661 jobs; 0 completed, 0 removed, 1387 idle, 267 running, 7 held, 0
suspended
```

Your job identification is a **$(ClusterID).$(ProcID):** 7952359.0

HTCondor: Submitting your job

Submit file. Executable, input, arguments, output, error and logs

- You **have to** specify the executable and optionally you can specify your input, output (stdout) and error (stderr) logs.
- Make sure that your script is correct (location, **permissions** and shebang)!
- You can set the global path. However, if no path is specified, the directory from which the job was submitted is used

```
executable = test.sh
input = input.txt
arguments = arg
output = test.out
error = test.err
log = test.log
```

PIC port d'informació científica

HTCondor: Submitting your job

Submit file. Requesting resources

- You can request cpu, gpu, memory and disk for your job using `request_cpus`, `request_gpus`, `request_memory` and `request_disk` respectively
- If request options are not present in your submit file, these default values are taken:
    - 1 cpu, 2 GB of memory per cpu, 15 GB of local disk per cpu and no GPUs

This job asks for 4 GB of RAM (use units!) and 8 CPUs

```
$ cat test-req.sub
executable = test.sh
arguments = -c 8 -t 60
output = test-req-$(ClusterId).$(ProcId).out
error = test-req-$(ClusterId).$(ProcId).err
log = test-req-$(ClusterId).$(ProcId).log

request_memory = 4 GB
request_cpus = 8

queue
```

```
$ condor_q 1462.0 -af RequestCpus RequestMemory
8 4096
```

Querying the resources requested, memory is in MB

PIC port d'informació científica

HTCondor: Submitting your job

Submit file. Requesting GPUs

- At PIC there are right now only 18 GPUs available
    - 8 GPUs for jupyter.pic.es (RTX 2080 Ti)
    - 8 GPUs for Magnesia group (V100)
    - 2 old GPUs for testing (GTX 1050 Ti)

- You can access any of them through HTCondor
    - However, RTX 2080 and V100 GPU work with **preemption enabled**: if a higher priority job needs the resources, your job will be killed and put in queue again
    - Contact us if you need to access to RTX2080 or V100 GPUs through HTCondor

PIC port d'informació científica

HTCondor: Submitting your job

Submit file. Requesting resources

- The default resources are 1 core and 2 GB of RAM. But you can access up to 8 cores and 32 GB without expecting long queue waits
  - You can access to more resources if needed, but first contact us
- The amount of resources requested determines the queue time: less resources, less time in queue
- Last 2 months Average Queue Time: 1.8 hours

HTCondor: Submitting your job

Submit file. The flavours

- 3 general flavours to limit the job’s walltime
  - short: 3 hours
  - medium: 48 hours
  - long: 96 hours
- Default flavour is **medium**
- When the job arrives to the time limit, it will be held and it remains in this status for 6 hours in the queue (`condor_q -af HoldReason`)
- Jobs that exceed 50% over the requested memory will be also held
- If you need more than 96 hours, consult us

```
$ cat test-flavour.sub
executable = test.sh
arguments = -c 1 -t 60
output = test-flavour-$(ClusterId).$(ProcId).out
error = test-flavour-$(ClusterId).$(ProcId).err
log = test-flavour-$(ClusterId).$(ProcId).log

+flavour="long"

queue
```

HTCondor: Submitting your job

Submit file. The queue statement

- `queue N`, to submit N number jobs
- Powerful command. Several ways to use queue. Example:
  - From file: Queue commands reads the information contained in a file

4 jobs submitted, considering we have multiple options in a list file arg_list.txt

```
$ cat test-queue3.sub
executable = test.sh
arguments = -c $(option1) -t $(option2)
output = test-queue3-$(ClusterId).$(ProcId).out
error = test-queue3-$(ClusterId).$(ProcId).err
log = test-queue3-$(ClusterId).$(ProcId).log

queue option1,option2 from args/arg_list.txt
```

```
$ cat arg_list.txt
1, 15
2, 10
1, 12
4, 13
```

HTCondor: Submitting your job

The priority and the Accounting Group

- The priority of your job is calculated depending on the Accounting Group
  - Your Accounting Group is defined by your primary group
  - HTCondor adds automatically the user to the Accounting Group
  - Each group have a quota of resources assigned to them

- If your primary group does not fit with the experiment you want to account for, you can add `+experiment="experiment"` in your submit file

```
$ cat test-experiment.sub
executable = test.sh
arguments = -c 1 -t 120
output = output-$(ClusterId).$(ProcId).out
error = error-$(ClusterId).$(ProcId).err
log = log-$(ClusterId).$(ProcId).log

+experiment="virgo"

queue
```

HTCondor: Submitting your job

Interactive submission

- There is the possibility to submit interactive jobs: `condor_submit -i/-interactive`

```
$ condor_submit -interactive
Submitting job(s).
1 job(s) submitted to cluster 7952079.
Waiting for job to start...
Welcome to slot1_6@tds228.pic.es!
You will be logged out after 7200 seconds of inactivity.
bash-4.2$
```

PIC port d'informació científica

HTCondor: Submitting your job

Use of containers

- The use of containers is always encouraged. Apptainer (old singularity) recommended
- You just need to add `+SingularityImage` to run inside a container
- All the shared filesystems are available inside the container

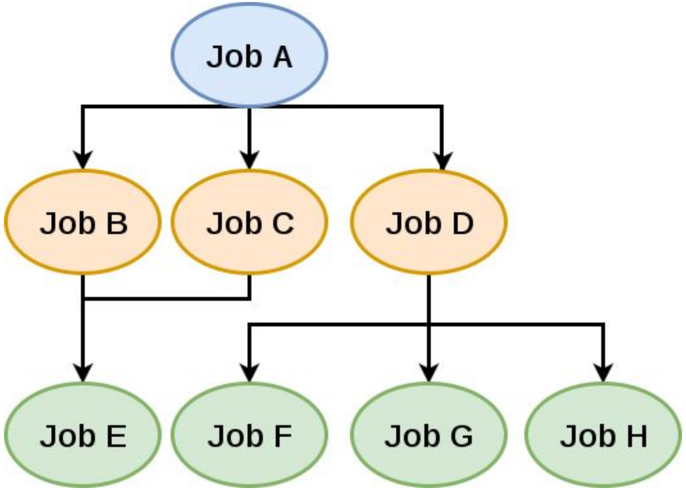
```
+SingularityImage = "/opt/apptainer-images/pic-centos7.sif"
```

HTCondor: Submitting your job

DAGMAN

- HTCondor Dagman (Directed Acyclic Graph Manager) is a meta-scheduler. You submit to the queue a scheduler that manages the execution order of several jobs

```
JOB   A   jobA.sub
JOB   B   jobB.sub
JOB   C   jobC.sub
JOB   D   jobD.sub
JOB   E   jobE.sub
JOB   F   jobF.sub
JOB   G   jobG.sub
JOB   H   jobH.sub
PARENT A CHILD B C D
PARENT B C CHILD E
PARENT D CHILD F G H
```



HTCondor: The generated files

The generated files

- `$_CONDOR_SCRATCH_DIR` is the directory where your job runs temporarily in the WN
  - All the data stored there is removed when the job finishes
  - It is recommended to avoid working directly in shared FS, work in the scratch of the node and copy at the end of your job if possible!
- The WNs have all your storage available, you can just decide how to move your data in your executable script
- The standard output and error are always transferred back when the job finishes (completed, removed or held)

HTCondor: Checking your job

condor_q

- allows constraints and several output formats
- Use `condor_q -l $ClusterId.$ProcId` to obtain all the job attributes

```
$ condor_q -const 'RequestCpus > 1 && JobStatus == 1' -nobatch
```

Use constraint (-const) to filter your jobs, remember that -nobatch shows the jobs ungrouped

```
-- Schedd: submit01.pic.es : <193.109.174.82:9618?... @ 03/14/19 09:42:46
 ID       OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
 630.0    cacosta         3/14 09:42   0+00:00:00 I  0    0.0 test.sh --cpu 1 --timeout 10s
 630.1    cacosta         3/14 09:42   0+00:00:00 I  0    0.0 test.sh --cpu 1 --timeout 10s
 630.2    cacosta         3/14 09:42   0+00:00:00 I  0    0.0 test.sh --cpu 1 --timeout 10s

Total for query: 3 jobs; 0 completed, 0 removed, 3 idle, 0 running, 0 held, 0 suspended
Total for cacosta: 7 jobs; 0 completed, 0 removed, 5 idle, 2 running, 0 held, 0 suspended
Total for all users: 7 jobs; 0 completed, 0 removed, 5 idle, 2 running, 0 held, 0 suspended
```

```
$ condor_q -const 'RequestCpus > 1 && JobStatus == 1' -nobatch -af ClusterId ProcId RequestCpus RequestMemory
630 0 4 2048
630 1 4 2048
630 2 4 2048
```

Autoformat (-af) allows you to control the format of condor_q output

HTCondor: Checking your job

condor_q -analyze/-better/-better-analyze

- allows you to check if there are WNs that can run your jobs

```
$ condor_q -better 7937412

[..]

         Slots
Step    Matched  Condition
-----  --------  ---------
[0]        1842  TARGET.WN_property == ifThenElse(MY.WN_property is undefined,"default",MY.WN_property)
[5]         423  TARGET.Disk >= RequestDisk
[6]         394  [0] && [5]
[7]           2  TARGET.Memory >= RequestMemory
[8]           0  [6] && [7]
[9]        2055  TARGET.FileSystemDomain == MY.FileSystemDomain

No successful match recorded.
Last failed match: Fri Oct  6 15:27:08 2023

Reason for last match failure: no match found

7937412.000:  Run analysis summary ignoring user priority.  Of 161 machines,
    161 are rejected by your job's requirements
      0 reject your job because of their own requirements
      0 match and are already running your jobs
      0 match but are serving other users
      0 are able to run your job

WARNING:  Be advised:
   No machines matched the jobs's constraints
```

PIC port d'informació científica

HTCondor: Checking your job

condor_history

- For finished and removed jobs that do not appear in the queue
- It allows similar constraints as condor_q (some of the attributes are slightly different, for instance `LastRemoteHost` for `RemoteHost`)
- Use `-limit N` to avoid very long queries
- You can check finished jobs up to 2 months ago

HTCondor: Checking your job

condor_rm

- Use `condor_rm` and the ClusterId and/or ProcId of your job
- It allows the use of constraints and "`-all`" to remove all your jobs

HTCondor: News and future

- Migration to AlmaLinux 9 OS in the coming month
    - Forced by the discontinuation of CentOs7
- Major upgrade from HTCondor 9 to 10
    - Shouldn't have an impact on the users
- You will receive new information soon
- Remember that other changes and new features will be in the wiki

Storage

Storage at PIC is separated in:

- Disk storage for internal access: **3.5 PB**
  - User Home. Small space, for scripts, logs, etc. Backups.
  - Software. Common space for group's users. Small space. Backups.
  - Common. Common data for members of a group. Backups.
  - Scratch. No critical data. No backups.
- Mass Storage for internal and external access:
  - Disk: **20 PB**. Tape: **63 PB**.
  - Local access on PICs resources (UIs, nodes)
  - Allow sharing data between centers/users.
  - Optional use of tape for long term data or backups.

IBM TS4500

Storage (II)

- User home
  - Quota / Reduced space
  - Only important data
- SW
  - Common software for all users
  - Small space
- Common
  - Common data
- Scratch
  - Not critical data (no backups)
  - Big space
  - No backups

NFS space for internal use

- Mounted on WN and UIs

- /pnfs
  - Different retention policies (Disk/Tape)
  - Not 100% Posix

Space for external use

- /pnfs (dCache)
- External access (https)
- Mounted on WN and UIs for local access

PIC port d'informació científica

Storage (backups)

Backup policies for disk internal storage

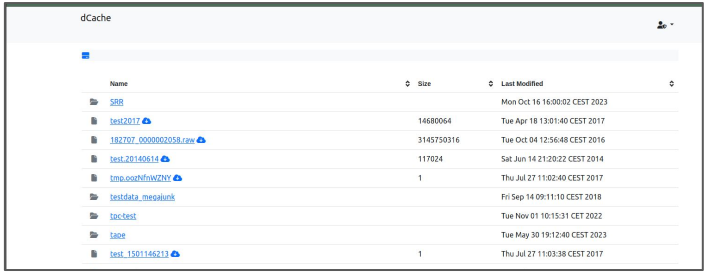
- Snapshots
  - Hourly (3 last snapshots) / daily (14) / weekly (3)
- Backup
  - Nightly copy to an external server

Endpoint	Snapshots	Backups
Home	yes	yes
Software	yes	yes
Common	yes	yes
Scratch	no	no

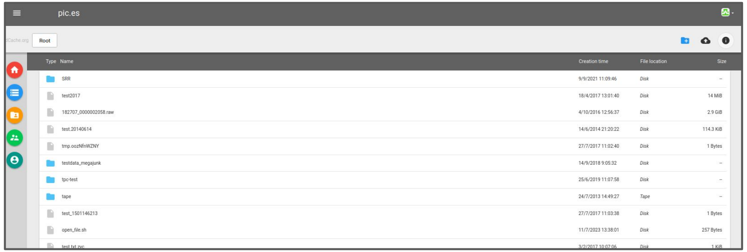
Storage (https access)

Web interface
/pnfs external access

- Webdav door
  - Read / Download files
  - Command line access (upload included)

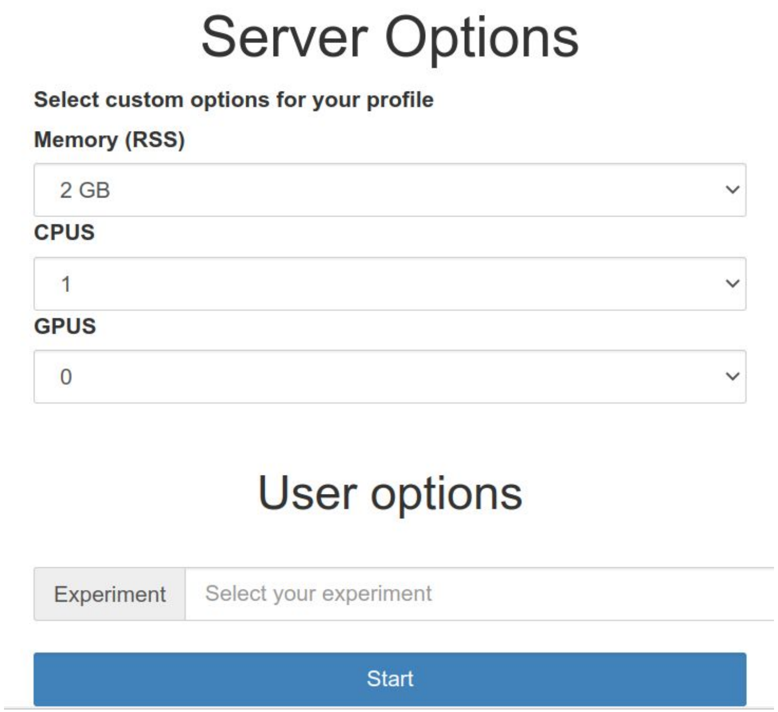


- Frontend dCacheView
  - Upload/download files

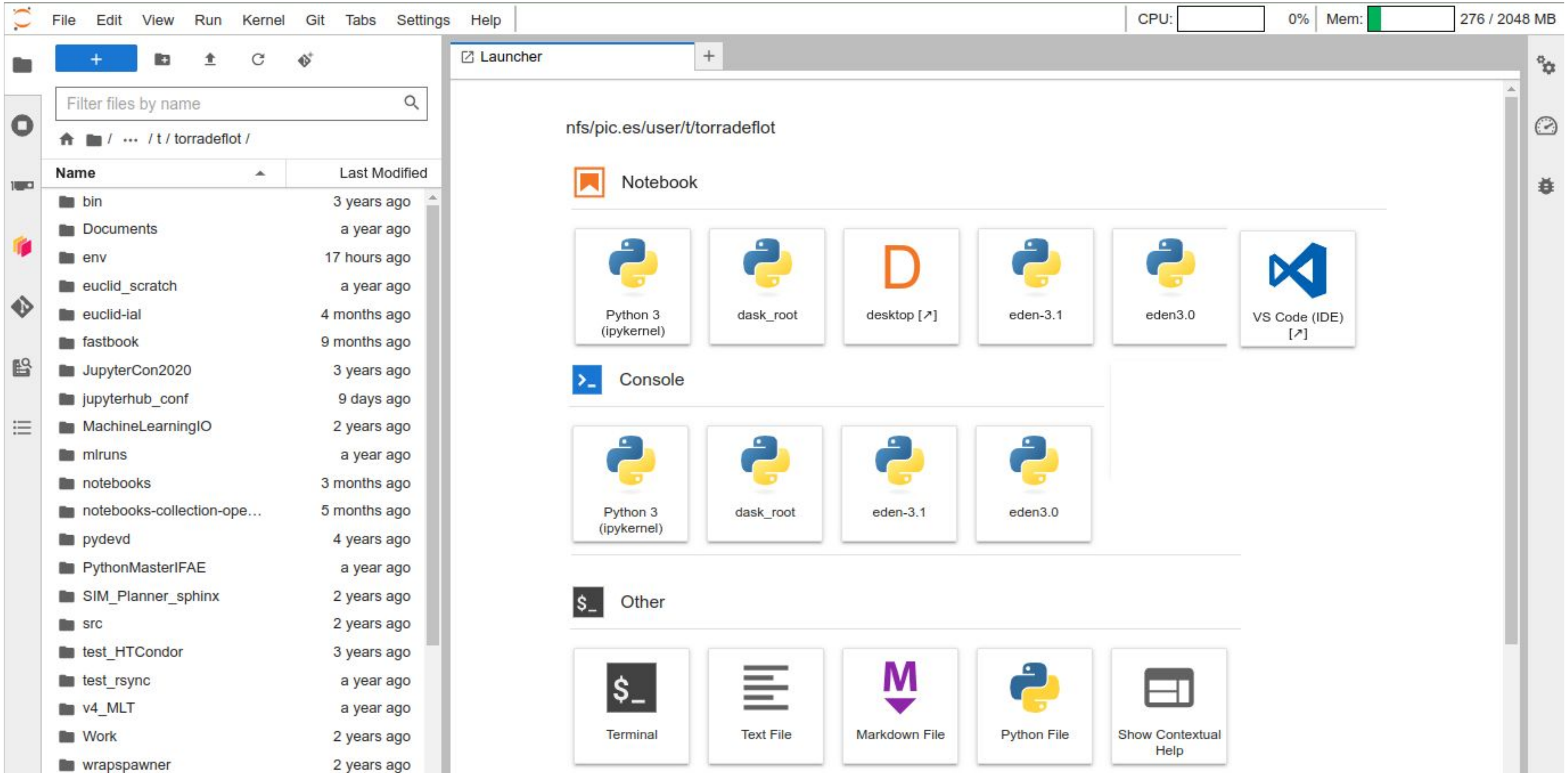


Jupyter: Introduction

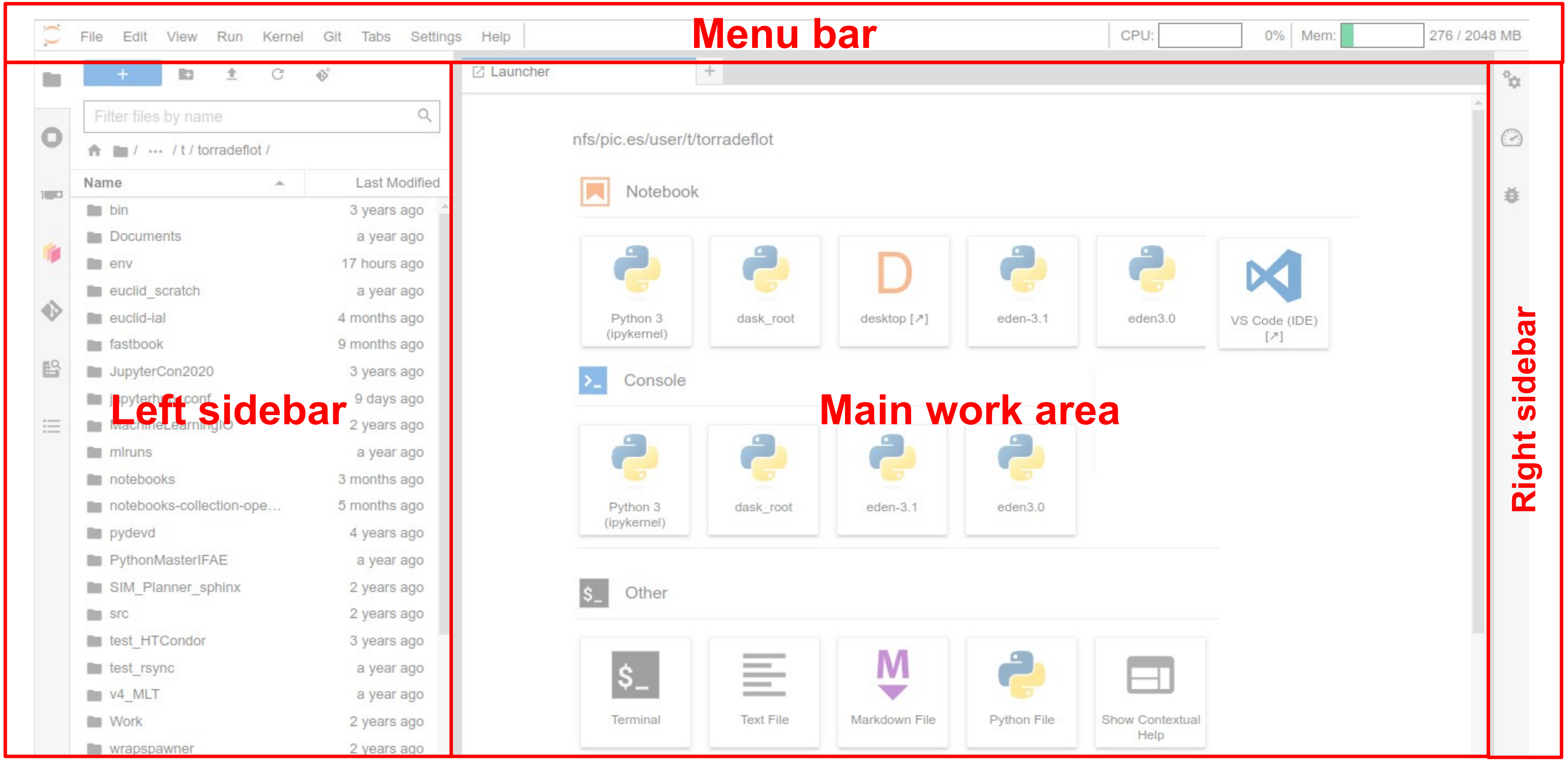
- jupyter.pic.es
- Launch a jupyter notebook server on PIC's HTC cluster
- User-defined resources
  - CPUs
  - Memory
  - GPUs
- Only a web browser
  and internet connection needed
- Access to PIC's massive storage
- No need to download data anymore !!



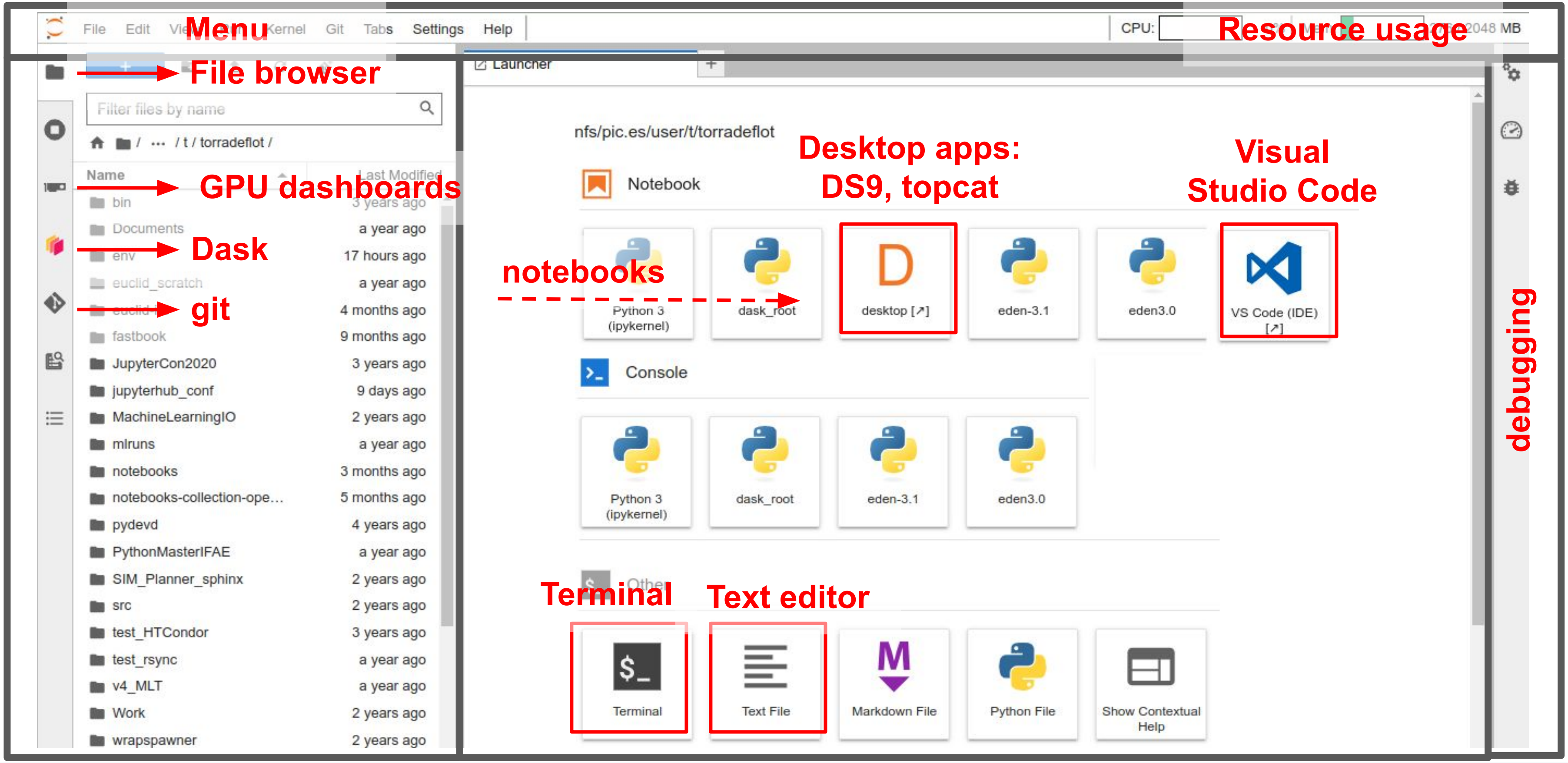
Jupyter: jupyterlab



Jupyter: jupyterlab

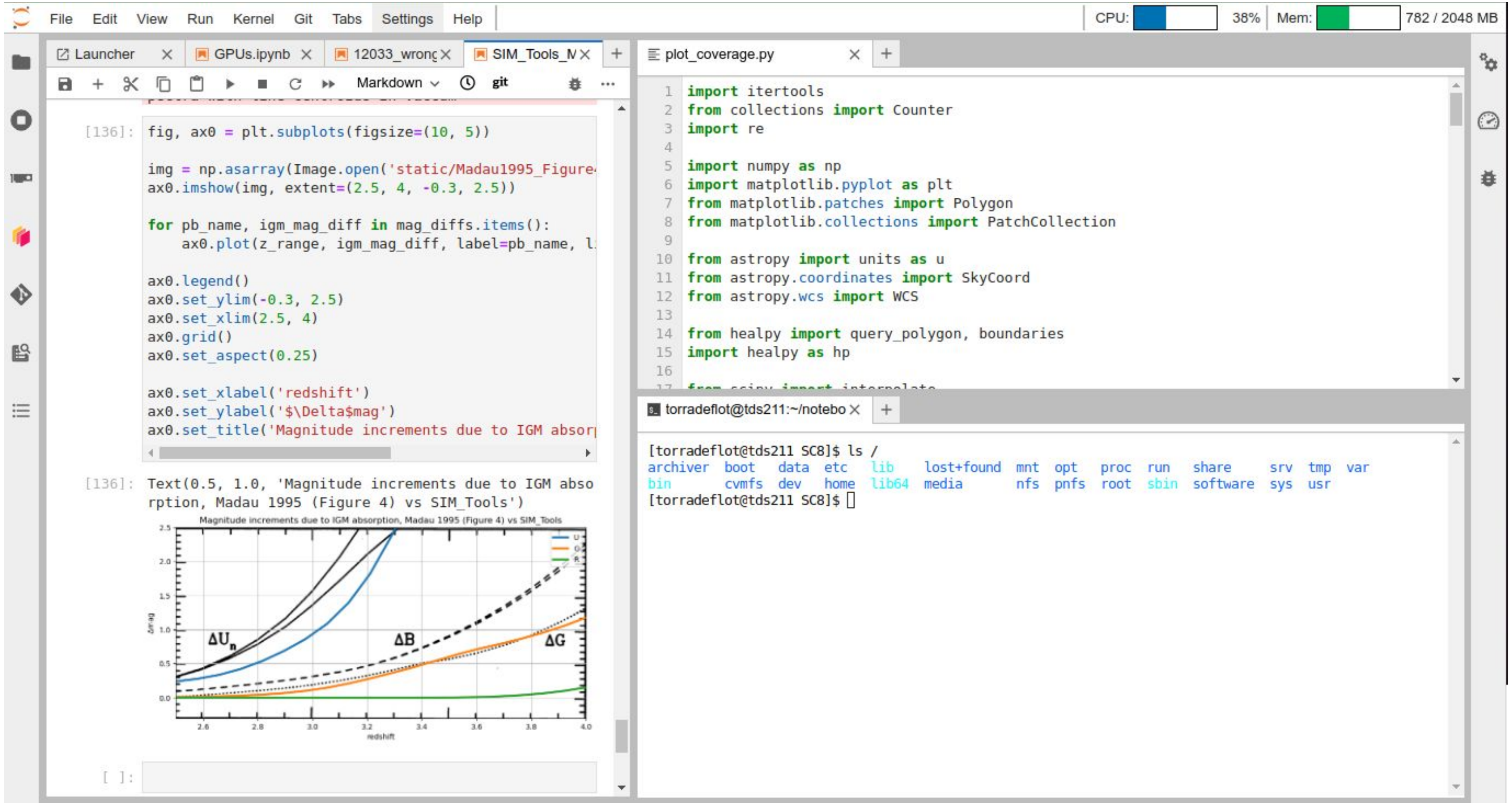


Jupyter: jupyterlab



PIC
port d'informació
científica

Jupyter: jupyterlab



PIC port d'informació científica

Jupyter: environments

We provide a python environment with the most common scientific libraries

Python 3.11 | Numpy 1.24 | Matplotlib 3.7 | pandas 2.0 | scipy 1.10

And some additions

astropy | scikit-learn | scikit-image | Dask | pillow

seaborn | bokeh | plotly | statsmodels | jupyter stack

Jupyter: environments

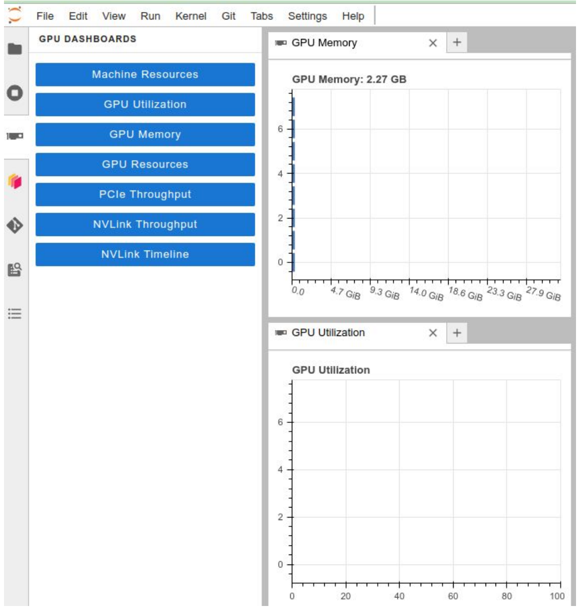
If this is not enough, you can still create custom environments and make them available to be used in your notebooks

```
> mamba create (-n {name} | -p {path}) ipykernel [pkg1 pkg2 …]
> mamba activate ({name} | {path})
> python -m ipykernel install –user –name {display name}
```

Detailed instructions here:
https://pwiki.pic.es/index.php?title=JupyterHub#Python_virtual_environments

Jupyter: GPUs

- GPUs available
  - gpu01: **8 x RTX 2080 Ti**, available via jupyter and HTCondor with preemption
  - gpu05: **8 x V100**, available via HTCondor with preemption, and a subset of 4 available via jupyter
  - gpu02 & gpu03: **1 x GeForce 1050 Ti**, available via HTCondor
- GPUs are a scarce resource. Don't request for a GPU unless you are really going to use it
- GPU dashboards in jupyterlab show the GPU usage
- No GPU libraries in the base environment



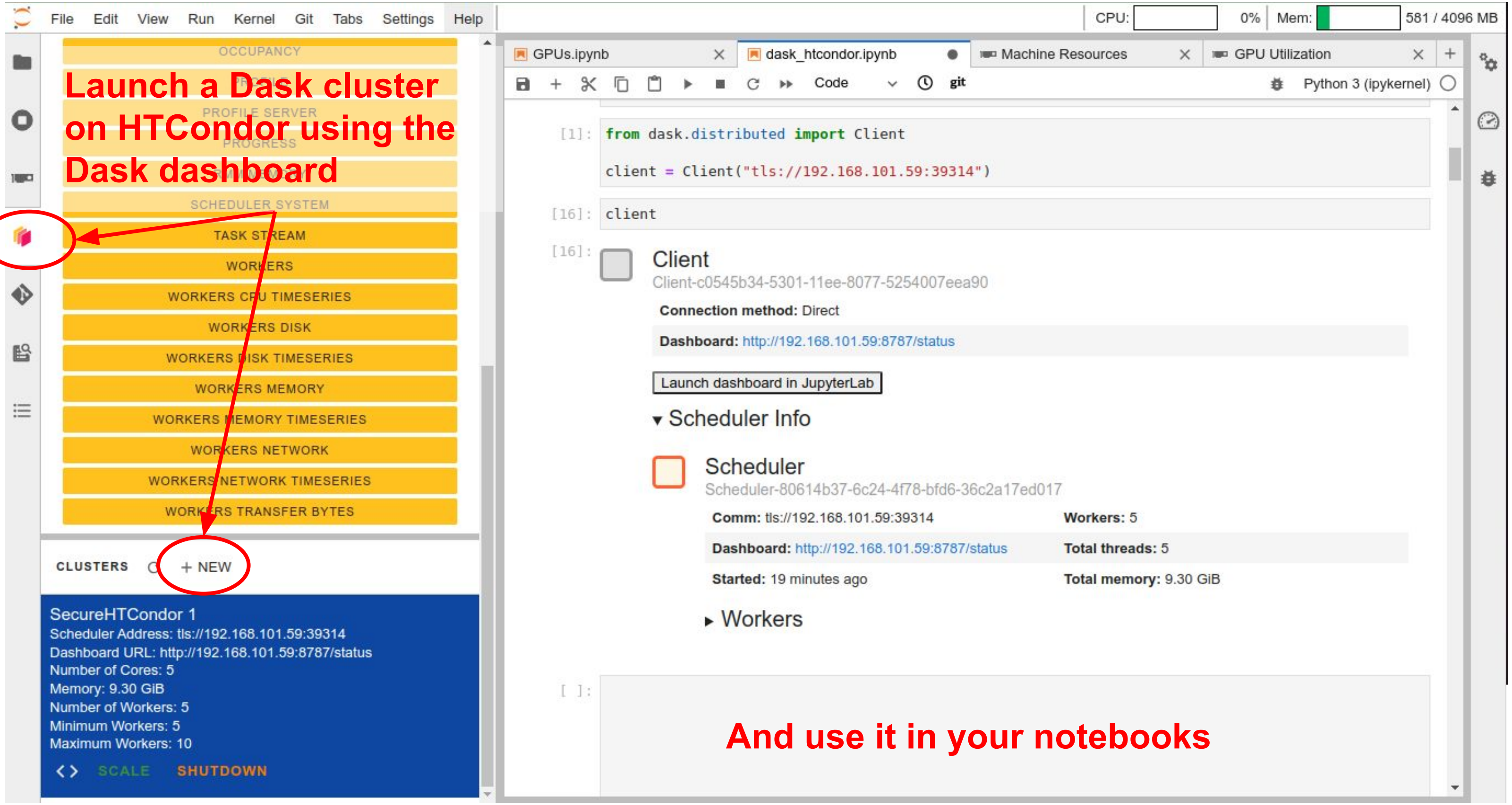
Jupyter: Dask

Dask provides the ability to scale Python data analytics to multiple machines

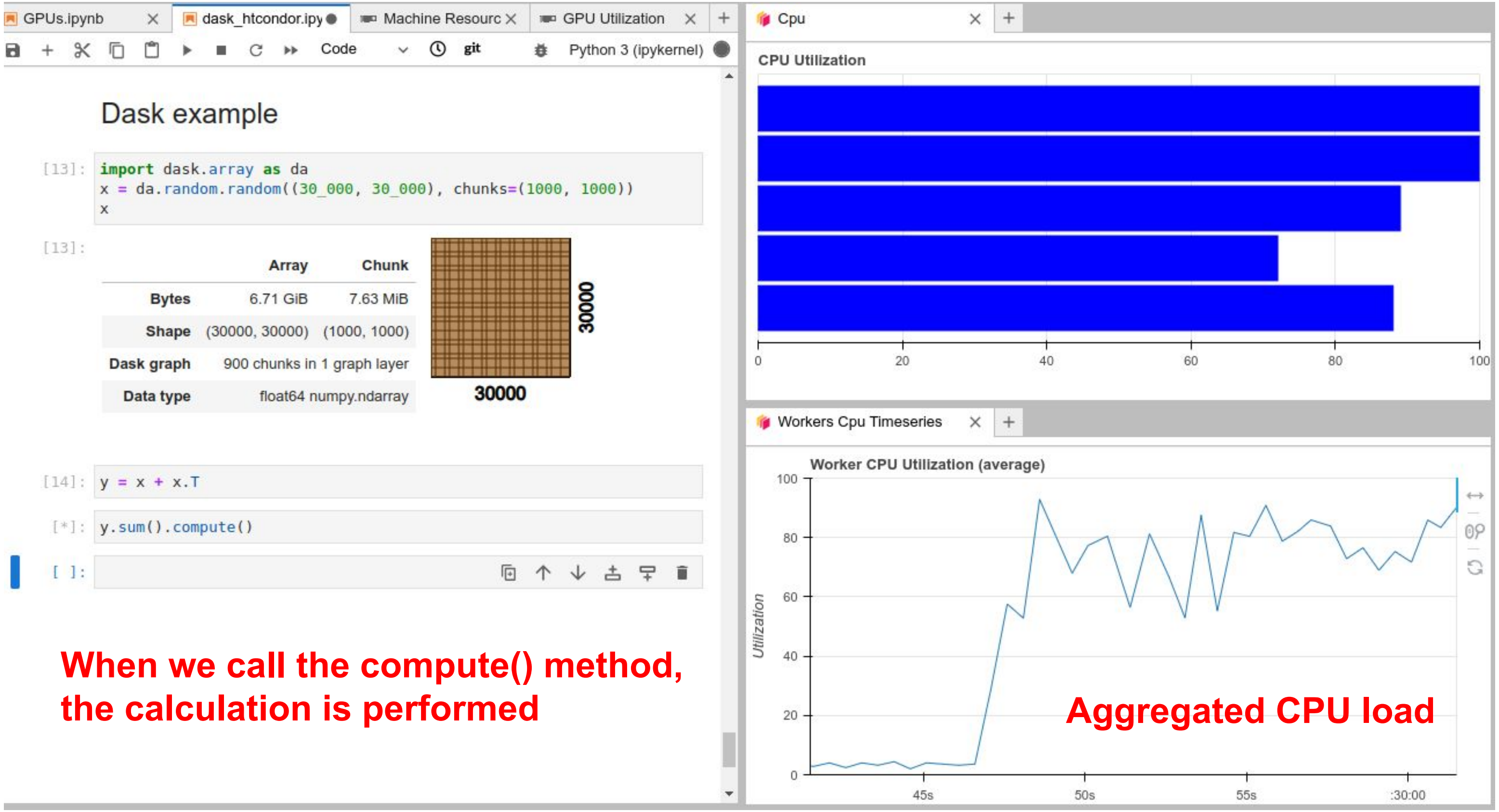
Why Dask?

- When going to large datasets, the analysis may not fit in a single machine
- Numpy and Pandas were not intended to scale to multiple machines
- Similar API (arrays and dataframes) but with lazy loading
- Can scale down to a single computer (e.g. for testing)
- Suited to scale up using PIC's HTC cluster (with your account limitations) using a lot of small workers

Jupyter: Dask



Jupyter: Dask



When we call the compute() method, the calculation is performed

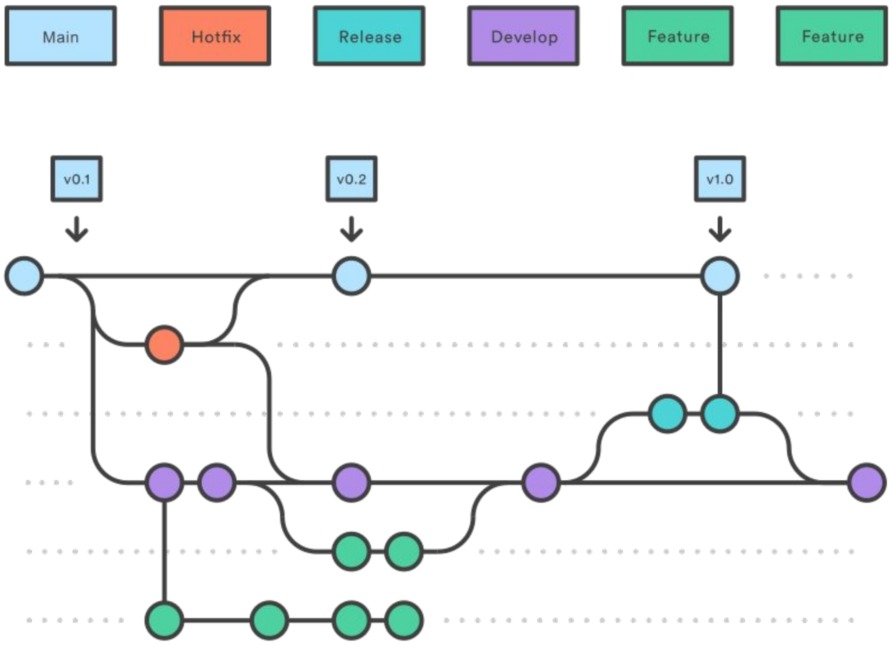
Aggregated CPU load

Gitlab: introduction

- Version Control System (VCS)
  - A tool to track and manage changes to digital assets (e.g. code)
  - Single source of truth
  - Enables parallel development / team collaboration
  - Full history
- Git
  - Started by Linus Torvald in 2005
  - Open source and free distributed VCS
  - Fast, Flexible, Secure … very good!
  - Has become the de facto standard
  - Is a MUST-have skill in software development
  - Has some learning curve
- Gitlab
  - Web-based Dev(Sec)Ops platform
  - Git repository manager
  - Many more features: CI/CD, issues, wiki

Gitlab: git overview

- Commit: snapshot of your code
- Branch: dynamic pointer to a commit
- Tag: immutable pointer to a commit



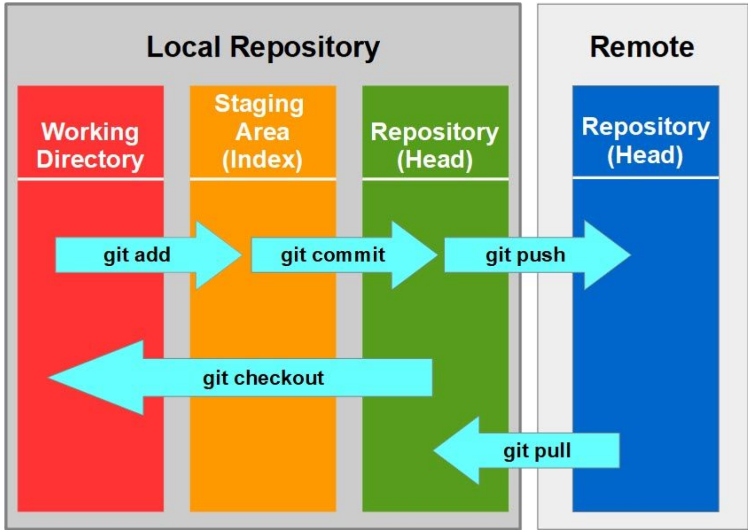
The gitflow branching strategy

Gitlab: git overview

PIC port d'informació científica

Commands

- `commit`: make a snapshot of your code
- `push`: upload your changes to the remote repository
- `checkout`: change your local copy to another snapshot
- `pull`: download changes from the remote repository
- `add`: select changes to be commited
- `clone`: create a local copy of a remote repository
- `merge`: add changes made in a snapshot to the current version of the code
- `log`, `branch`, `status`, `diff`, `config`, `init`, etc

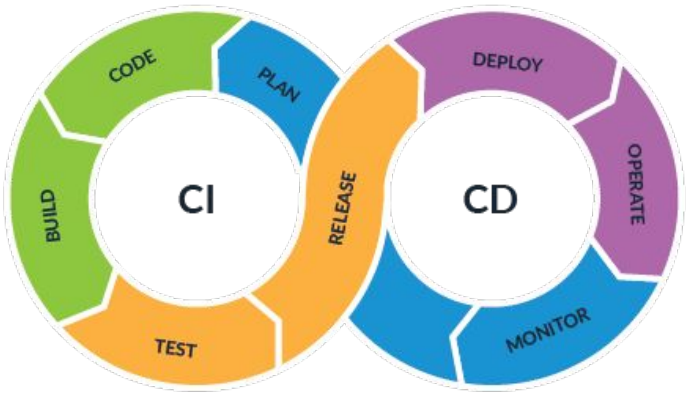


Gitlab: git best practices / suggestions

- General rules
  - **Make small and atomic changes**. Commit/push often.
  - **Use branches** for dedicated/long developments
  - **Keep the main branch stable**: the tests should always pass (yes you should have tests!!)
  - **Write descriptive commit messages**: avoid messages like “changes”, “test” or “.”
  - **Adopt a branching strategy**: gitflow, trunk-based,...
  - **Do code reviews** if feasible
- Our contributions
  - **Use git**. If you work alone or in a team, use it!
  - **Do not upload big binary files** to a git repository. Git is not for data, it is for code.
  - **Use .gitignore** to track only relevant files
  - **Do not track jupyter notebooks** (.ipynb) directly, pair them with a script
  - **Do not upload confidential data (NEVER!)** passwords, ssh keys, etc will be there forever

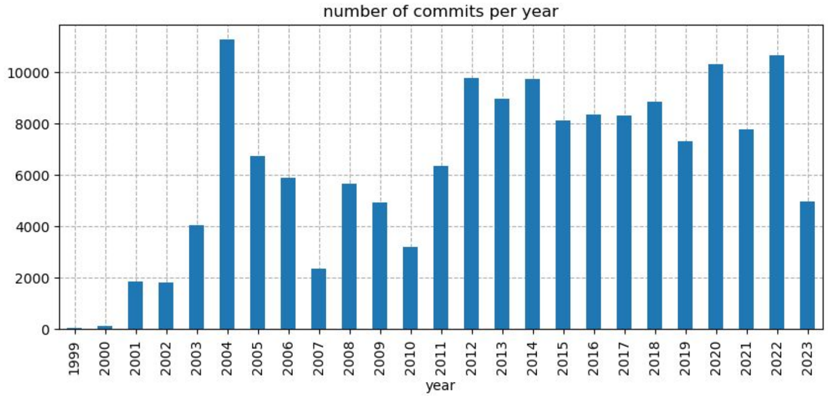
Gitlab: CI/CD

- Continuous Integration / Continuous Deployment
- Automate the stages of software development, that come after writing the code
  - Build
  - Test
  - Deploy
- When should you use it?
  - Whenever you want to automate some of these steps
  - Nearly always



PIC's Gitlab in numbers

- Already being actively used by many groups / users
  - 795 projects
  - 82 groups
  - 358 users
  - 160k commits
  - 700 issues
  - 1100 MRs
- CI /CD
  - ~6.4k pipelines
  - ~22k jobs
- Hardware
  - VM with 4 cores and 8GB RAM
  - 75GB of disk

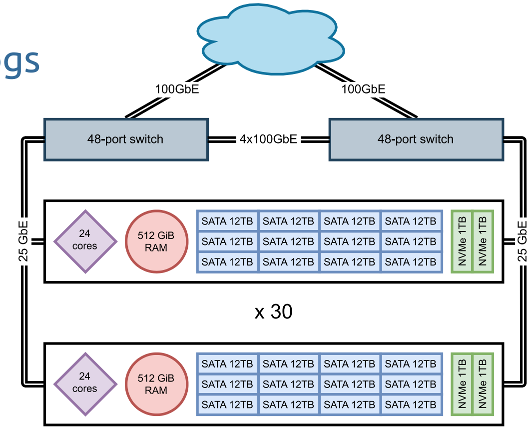


Group	N pipelines
IFAE websites	2318
Magic	762
Hadoop (PIC)	598
Virgo SW	565
Rucio (PIC)	488

Big Data service

PIC port d'informació científica

- hadoop cluster
  - 30 nodes, 720 cores, 15 TiB RAM, 60 TiB NVMe cache, **2.5 PiB net storage**
    - **⅔ in production next month**
  - Based on a custom developed in-house Hadoop distribution
- CosmoHub (based on HIVE)
  - Interactive exploration and distribution of astronomical catalogs
    - Suited for very large tables (>100 million rows)
  - User Defined Functions
  - Bring your own data!
- Apache Spark
  - Parallel processing framework (similar to Dask)
  - Supports Python, Scala and R
  - Oriented towards accessing/processing catalogs stored on Hive



CosmoHub


Login / Sign Up
COSMO HUB
Build your own Universe
Interactive data analysis of massive cosmological data without any SQL knowledge
Billions of observed and simulated galaxies
Superfast queries means superfast results
Features to make you work faster and easier
Online plotting preview and data download
Learn more

User Defined Functions (UDFs)

- HEALPix
  - Following healpy calling semantics BUT using resolution **order** instead of nside
  - Implemented functions:
    - Pixel conversion: `ang2pix, ang2vec, pix2ang, pix2vec, vec2ang, vec2pix`
    - Ordering conversion: `nest2ring, ring2nest`
    - Other: `angdist, neighbours, npix2nside, nside2npix, nside2order`
- Array
  - To aggregate on equinumerous array columns (e.g. spectra, pdf, ...)
  - Implemented functions:
    - `array_min, array_max, array_count, array_sum`
    - `array_avg, array_stddev_pop, array_stddev_samp, array_var_pop, array_var_samp`
- Geometric (ADQL)
  - Deal with spherical geometries: POINT, CIRCLE, BOX/POLYGON and REGION
  - Implemented functions:
    - Initialize geometries: `adql_point, adql_circle, adql_box, adql_polygon, adql_region`
    - Basic operations: `adql_area, adql_centroid, adql_coord1, adql_coord2, adql_distance`
    - Set operations: `adql_contains, adql_intersects, adql_complement`
    - Aggregations: `adql_union, adql_intersection`

COSMO HUB

~ 150 active users

~ 13K custom catalogs

~ 40 TiB hosted data

> 100 catalogs

Public catalogs

- Gaia (DR3, Mean Spectrum, EDR3, DR2 & DR1)
- DESI Legacy Survey (DR9, DR8 PZ)
- DESI Legacy Survey with Photoz (DR8)
- COSMOS 2020 (Classic | Farmer)
- COSMOS 2015 Laigle (v2.1)
- LSST DESC DC2 (Truth-match | Object table)
- DES DR2
- DES Y1A1 Morphological catalog (v1.0)
- DES Y1A1 Gold Data (v1.0)
- GLADE (v2.3, v2.4) & GLADE +
- VIPERS photometry and spectroscopy (PDR2)
- KiDS (DR4)
- CANDELS Bulge-Disk decomposition (2018)
- CFHTLenS (good fields) (v1.2)
- Alhambra photometric redshifts (v1.0)
- ALHAMBRA S/G CLASSIFIED (v1.0)
- PAUS+COSMOS photo-z catalog (v0.4)
- PAUS-COSMOS Early Data Release (v1.0)
- PAU.MillGas Lightcone (2016-07-18)
- DEEP2 Redshift catalog (DR4)
- MICE halo properties
- MICECAT (v2.0, v1.0)

Bring your own data!!!

How to upload new catalogs into Hive/CosmoHub

- Catalog data
  - Download into any PIC storage (pnfs, scratch)
  - Or provide clear download information (URL, path and credentials)
  - Preferred formats, in order (Parquet, FITS, CSV)

- Metadata
  - For each catalog
    - provide a name, version, short and long description, in Markdown
    - specify whether this data is simulated or comes from observations
    - list which groups/projects have access, or public otherwise
  - For each column, specify its data type, units and description, in plain text.

- Parallel processing framework
  - 3 compatible APIs
    - SQL
    - Dataframes
    - RDD
  - Interfaces with Hive/CosmoHub tables
  - Can also access massive storage (PNFS/Ceph/NFS)
  - Dual execution: notebook and batch

```
df = spark.sql("""
  SELECT id, ra, dec
  FROM cosmohub.micecatv1_0_hpix
  LIMIT 100
""")
df
```

```
DataFrame[id: int, ra: double, dec: double]
```

```
df.show(5)
```

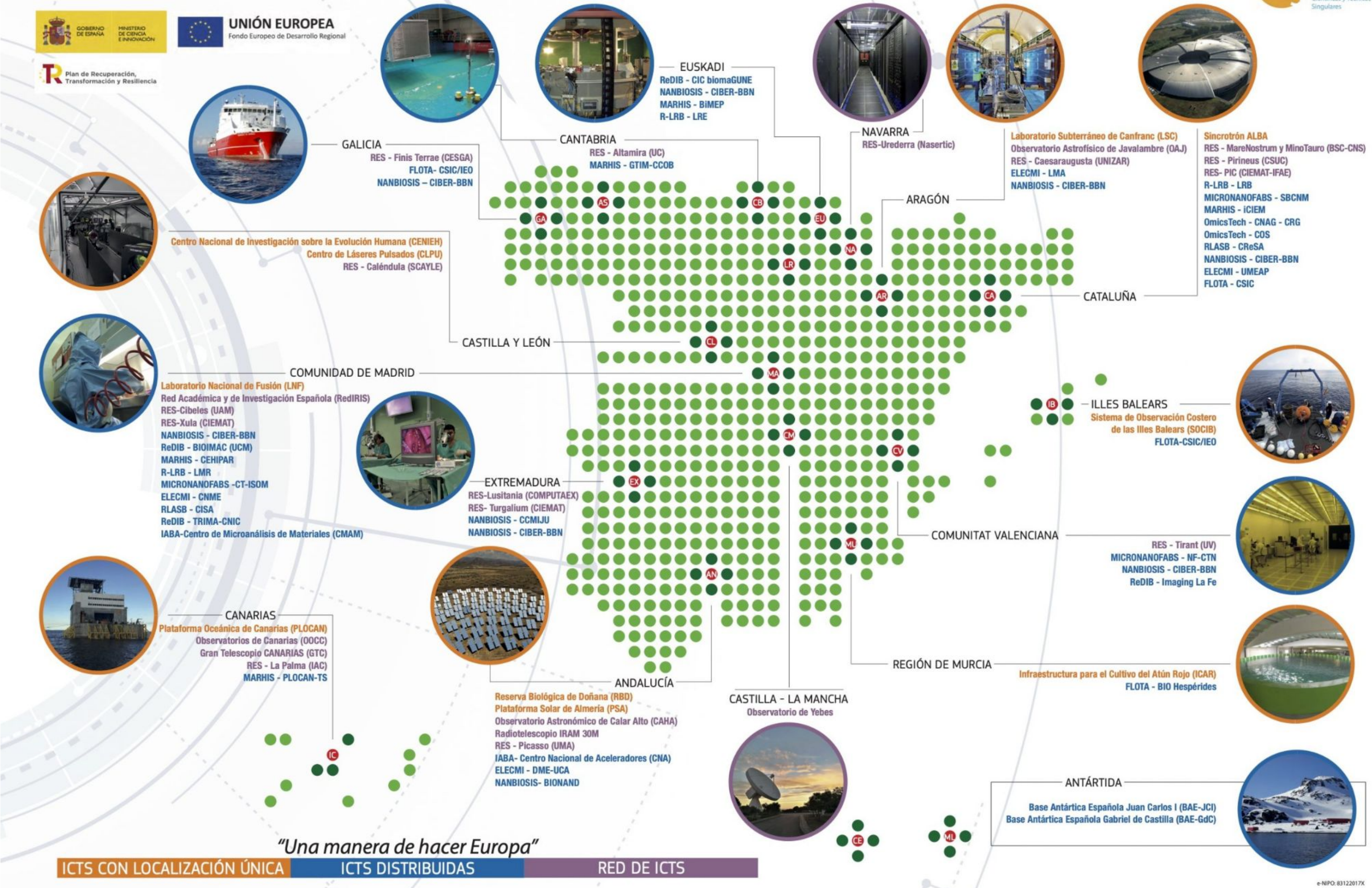
```
+---------+---------+---------+
|       id|       ra|      dec|
+---------+---------+---------+
|191225057|18.523232|79.887398|
| 49810401|59.949303|20.816753|
|  9887201| 22.78075|46.971172|
| 11503841|51.193577| 17.27203|
| 43089377| 8.418952|16.733221|
+---------+---------+---------+
only showing top 5 rows
```

Backup slides

PIC - Barcelona

Port d'Informació Científica

MAPA DE INFRAESTRUCTURAS CIENTÍFICAS Y TÉCNICAS SINGULARES



PIC
port d'informació
científica

PIC services

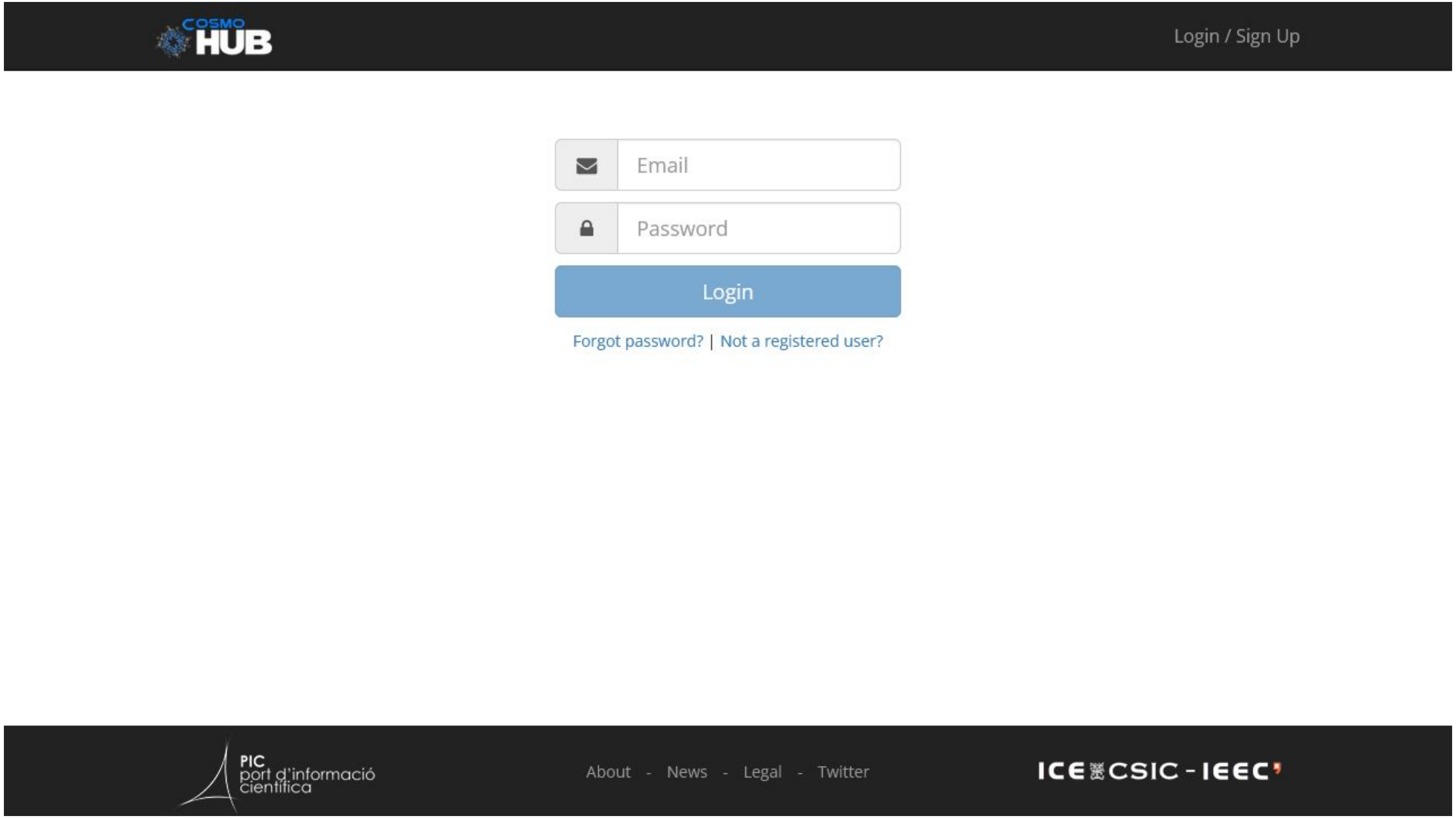
- Computing, data processing and analysis
  - Batch processing through HTCondor
  - JupyterHub
- Mass storage (tens of Petabytes)
  - Tape
  - Disk
- Big Data - Hadoop Cluster
  - CosmoHub
  - Spark Notebooks*
- Web Services (Gitlab, Wiki, Redmine, Webdav, Monitoring, etc.)
- Consulting support

PIC port d'informació científica

Big Data service

- New cluster
  - 30 nodes, 720 cores, 15 TiB RAM, 60 TiB NVMe cache, 2.5 PiB net storage
  - Based on a custom developed in-house Hadoop distribution
  - (intro hadoop, replica, ec, hdfs, yarn)
- Hive
  - Data warehouse based on distributed architecture
  - Better suited for very large tables (>100 million rows, >100 columns)
- CosmoHub
  - Custom web interface in front of Hive
  - Interactive plots
  - Download custom subsets
  - Standard formats: CSV, FITS, ASDF, Parquet
  - Additional functions: HEALPix, Array, Geometric
- Spark

Login/Register



PIC port d'informació científica

Catalog list

COSMO HUB Help Catalogs Activity tallada@pic.es

gaia

Name	Version	Description	Origin	Date
Gaia mean spectrum	DR3	RVS_MEAN_SPECTRUM	Observed	2023-02-14
Gaia	DR3	Gaia Data Release 3	Observed	2022-06-13
Gaia	EDR3	Gaia Early Data Release 3	Observed	2020-12-03
Gaia	DR2	Gaia Data Release 2	Observed	2018-04-25
Gaia	DR1	Gaia Data Release 1	Observed	2016-12-15

PIC port d'informació científica About - News - Legal - Twitter ICE CSIC - IEEC

PIC port d'informació científica

Catalog description

COSMO HUB Help Catalogs Activity tallada@pic.es

Gaia DR3 Columns Sampling Filters Query Analysis Format Request

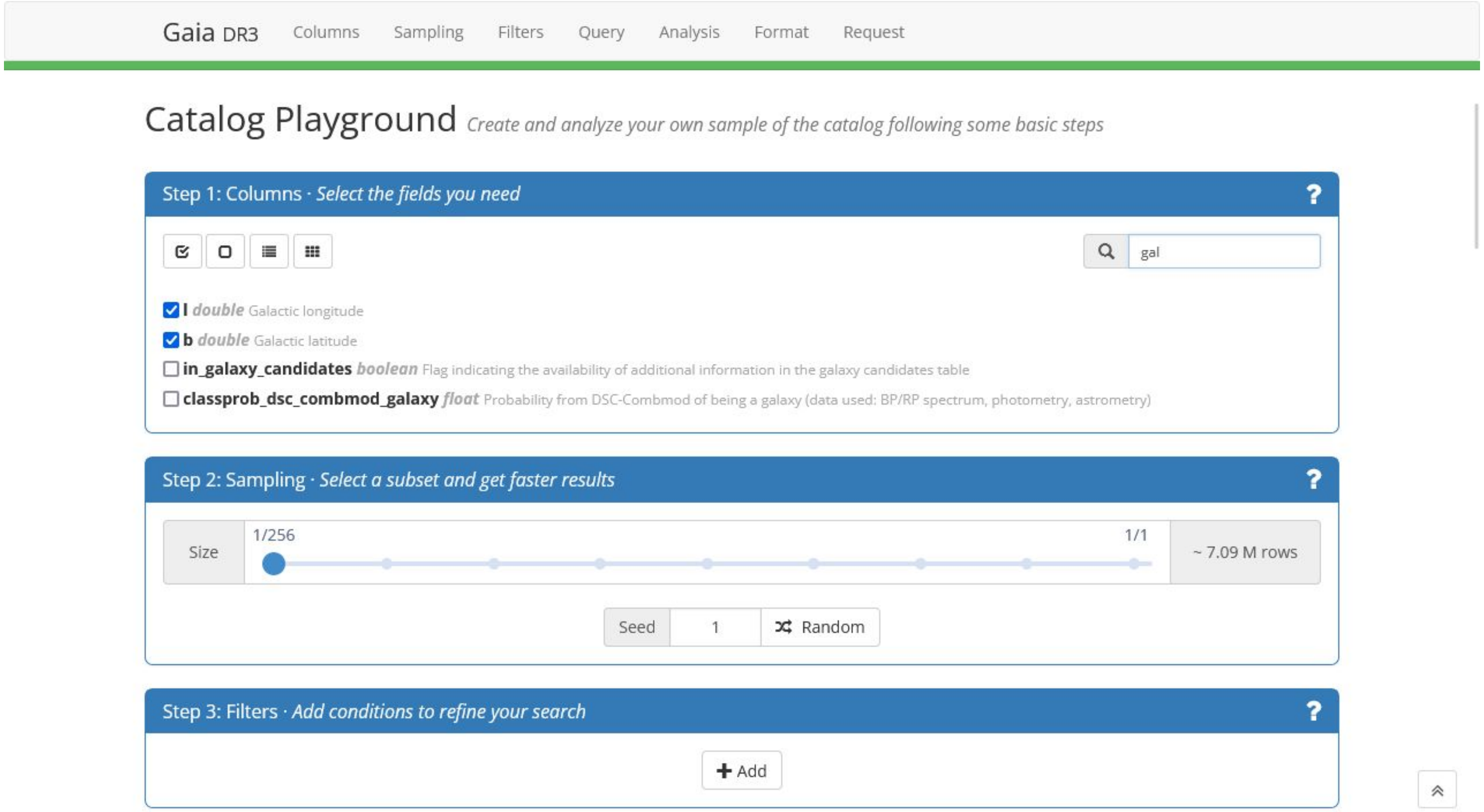
The third Gaia data release (Gaia DR3) consists of the set of data released as Gaia Early Data Release 3 on 3 December 2020 complemented with new data released on 13 June 2022. The set of data released as Gaia Early Data Release 3 (Gaia EDR3) on 3 December 2020 comprises:

- The full astrometric solution — positions on the sky (α, δ), parallaxes, and proper motions — for around 1.46 billion (1.46 10^9) sources, with a limiting magnitude of about G ≈ 21 and a bright limit of about G ≈ 3. The astrometric solution is accompanied with some new quality indicators, like RUWE, and source image descriptors.
- The full astrometric solution has been done as 5-parameter solution for 585 million sources and as 6-parameter solution for 882 million sources. In the 6-parameter solution, the additional fitted quantity is the so-called pseudo-colour that had to be included for sources without high-quality colour information.
- In addition, two-parameters solutions - positions on the sky (α, δ) - for around 344 million additional sources.
- G magnitudes for around 1.806 billion sources (with the known issue present in EDR3 corrected in Gaia DR3).
- G_BP and G_RP magnitudes for around 1.54 billion and 1.55 billion sources, respectively.

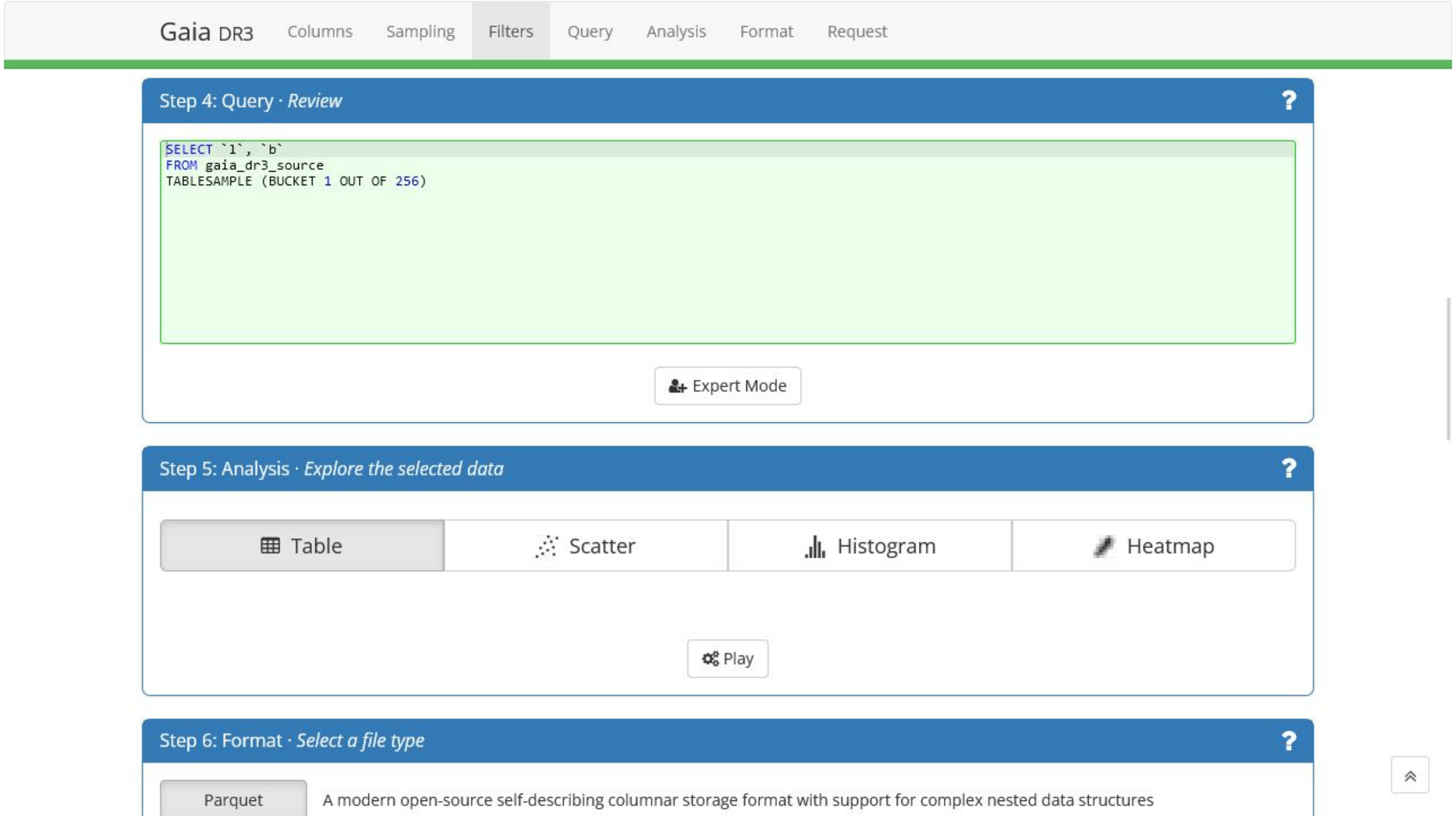
In Gaia Data Release 3 (Gaia DR3), the above set of data is complemented with new products released on 13 June 2022:

- Object classifications for 1.59 billion sources and astrophysical parameters (T_eff, logg, [M/H], A_G, distance, etc.) from BP/RP spectra for 470 million objects, including MCMC samples for most sources with astrophysical parameters. Other astrophysical parameters from the BP/RP spectra include:
  - Spectral types (217 million stars) and emission-line star classifications (57,000 stars);
  - Spectroscopic parameters for 2.3 million hot stars, 94,000 ultra-cool stars, activity index for 1.3 million cool stars, and H-alpha emission for 235 million stars;
  - Evolutionary parameters (mass and age) for 128 million stars;
  - Astrophysical parameters for 348 million objects based on the assumption of an unresolved binary in the BP/RP spectra;
  - Self-organised map (outlier) analysis based on 56 million sources with the weakest object classifications.
- Astrophysical parameters (T_eff, logg, [M/H], [X/M] for 12 elements, etc.) from RVS spectra for 5.5 million objects, including diffuse interstellar bands for 472,000 objects.
- All-sky total galactic extinction maps at 4 different spatial resolutions (HEALPix levels 6, 7, 8, and 9).
- Mean BP/RP spectra for 219 million sources, most of them with G < 17.6 mag.
- Mean RVS spectra for 1 million well-behaved objects.
- Mean radial velocities for 33 million stars and mean G_RVS magnitudes for 32 million objects with G_RVS <~ 14 mag with effective temperatures (T_eff) in the range of ~3100 to 14,500 K.

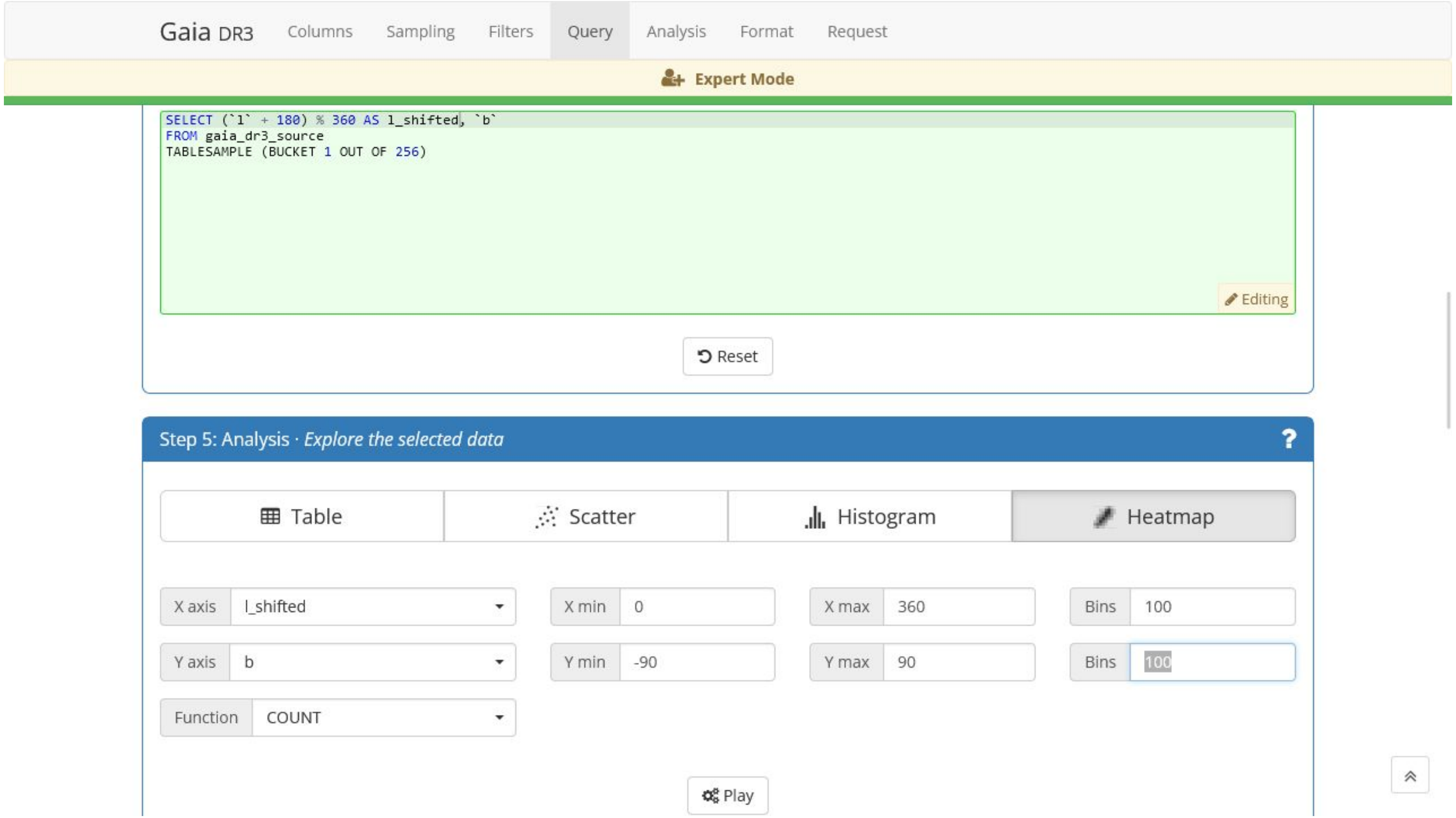
Columns, Sampling and Filters



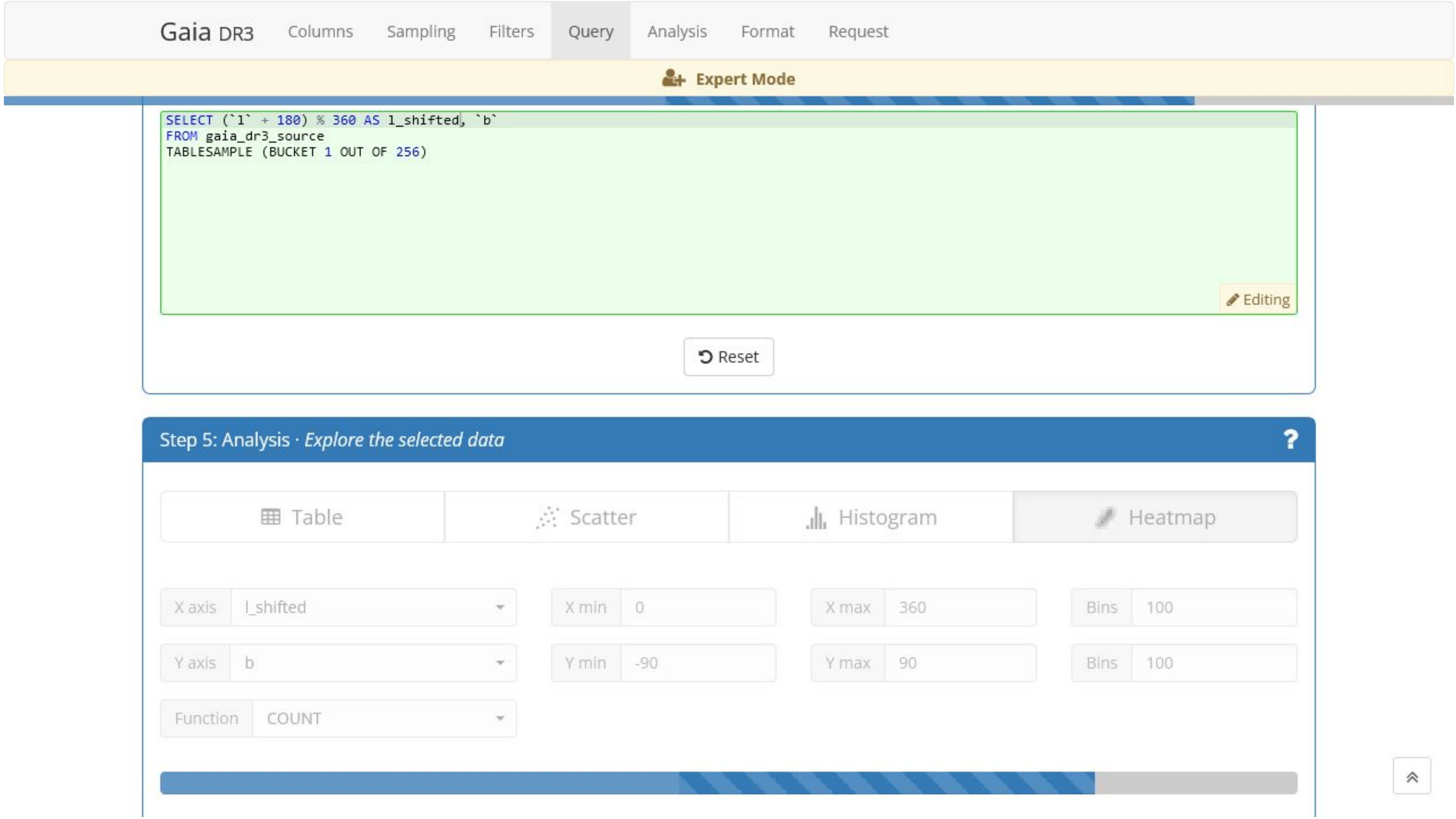
SQL view, interactive plots



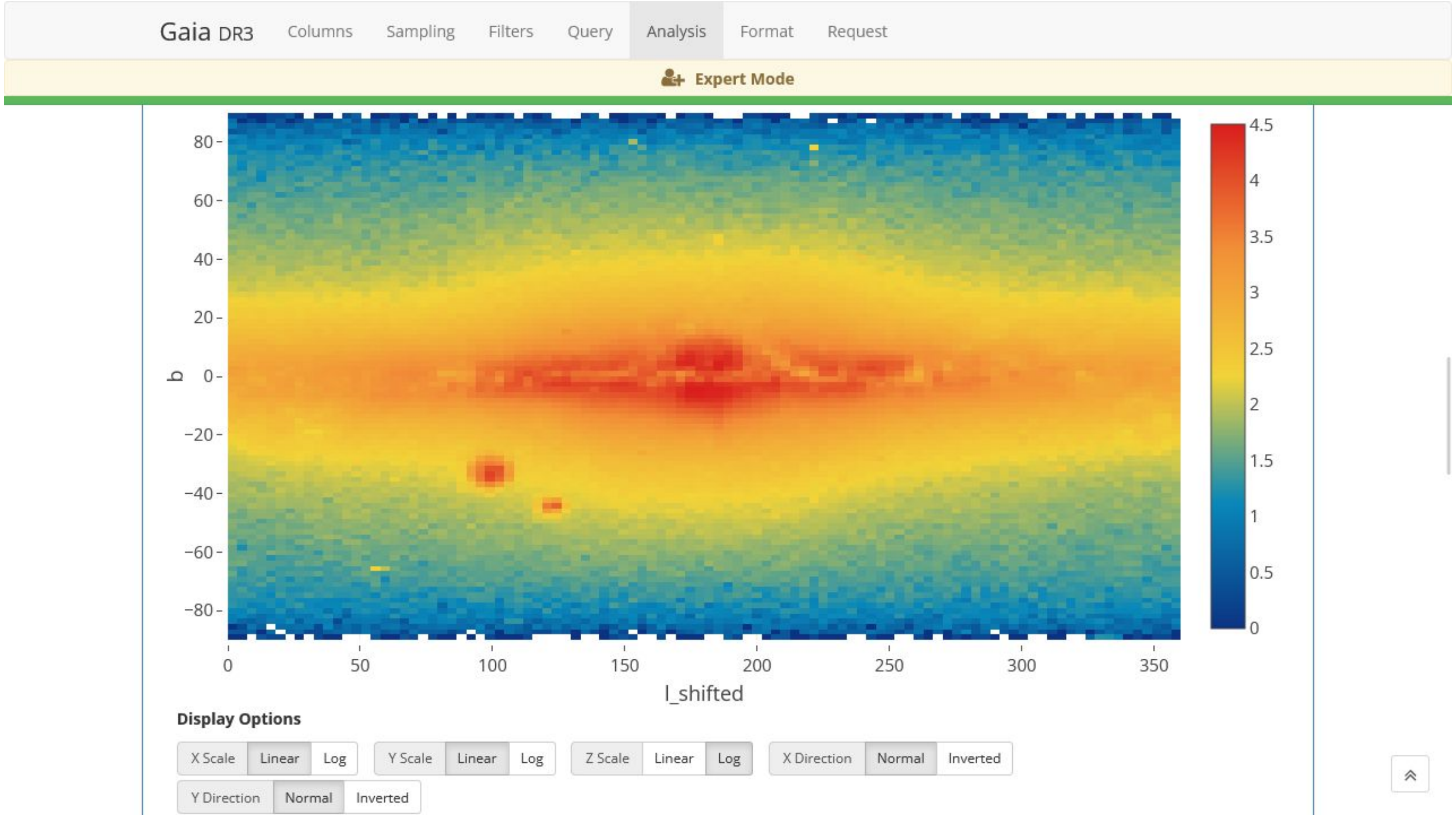
SQL editor, plot configuration



Query execution progress



Heatmap plot



PIC
port d'informació
científica

Output format and Citation info (I)

Gaia DR3 | Columns | Sampling | Filters | Query | Analysis | Format | Request

Expert Mode

Step 6: Format · *Select a file type*

Parquet	A modern open-source self-describing columnar storage format with support for complex nested data structures
CSV.BZ2	Bzip2 compressed Comma-separated values file (please check Help #4 if using Pandas DataFrame)
FITS	The Flexible Image Transport System is the most commonly used digital file format in astronomy
ASDF	The Advanced Scientific Data Format pretends to be the successor for the immensely successful FITS format

Step 7: Request · *Review citation guides*

How to cite CosmoHub

If you have used in your work any plots or data produced through CosmoHub please include in the Acknowledgments section the following snippet:

This work has made use of CosmoHub.

CosmoHub has been developed by the Port d'Informació Científica (PIC), maintained through a collaboration of the Institut de Física d'Altes Energies (IFAE) and the Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas (CIEMAT) and the Institute of Space Sciences (CSIC & IEEC), and was partially funded by the "Plan Estatal de Investigación Científica y Técnica y de Innovación" program of the Spanish government.

In addition, please cite the following publications:

Tallada et al. 2020 *CosmoHub on Hadoop: Interactive analysis and distribution of cosmological data* | PDF - BibTeX

Carretero et al. 2018 *CosmoHub and SciPIC: Massive cosmological data analysis, distribution and generation using a Big Data platform* | PDF - BibTeX

Output format and Citation info (II)

PIC port d'informació científica

Gaia DR3 | Columns | Sampling | Filters | Query | Analysis | Format | Request

Expert Mode

- Gaia Collaboration et al. (2022g): Gaia EDR3: The celestial reference frame (Gaia-CRF3);
- Lindegren et al. (2020): Gaia EDR3: Parallax bias versus magnitude, colour, and position;
- Rowell et al. (2020): Gaia EDR3: Modelling and calibration of Gaia's point and line spread functions;
- Marrese et al. (2022): Gaia EDR3: Cross-match with external catalogues – Algorithms and results.

For reference, the following papers describe the performance verification that DPAC has performed on Gaia DR3:

- Gaia Collaboration et al. (2022e): Gaia DR3: Mapping the asymmetric disc of the Milky Way;
- Gaia Collaboration et al. (2022d): Gaia DR3: Pulsations in main-sequence OBAF stars as observed by Gaia;
- Gaia Collaboration et al. (2022f): Gaia DR3: Reflectance spectra of solar system small bodies;
- Gaia Collaboration et al. (2022h): Gaia DR3: The Galaxy in your preferred colours – synthetic photometry from Gaia low-resolution spectra;
- Gaia Collaboration et al. (2022a): Gaia DR3: Stellar multiplicity – a teaser for the hidden treasure;
- Gaia Collaboration et al. (2022b): Gaia DR3: The extragalactic content;
- Gaia Collaboration et al. (2022i): Gaia DR3: Chemical cartography of the Milky Way;
- Gaia Collaboration et al. (2022c): Gaia DR3: The golden sample of astrophysical parameters;
- Gaia Collaboration et al. (2022j): Gaia DR3: Exploring and mapping the diffuse interstellar bands at 862 nm.

The Gaia data are open and free to use, provided credit is given to 'ESA/Gaia/DPAC'. In general, access to, and use of, ESA's Gaia archive (hereafter called 'the website') constitutes acceptance of the following, general terms and conditions. Neither ESA nor any other party involved in creating, producing, or delivering the website shall be liable for any direct, incidental, consequential, indirect, or punitive damages arising out of user access to, or use of, the website. The website does not guarantee the accuracy of information provided by external sources and accepts no responsibility or liability for any consequences arising from the use of such data.

How to distribute this catalog

The Gaia data are open and free to use, provided credit is given to 'ESA/Gaia/DPAC'.

☑ I have read the instructions on how to cite CosmoHub and this catalog in my publications.

✔ Request

PIC
port d'informació científica

Activity panel

COSMO HUB Help Catalogs Activity tallada@pic.es

Search...

ID	Query	Date	Status	Results
14424	SELECT CAST(`kind` AS FLOAT), CAST(`x_gal` AS FLOAT), -- x CAST(`y_gal` AS FLOAT), -- y CAST(`z_gal` AS FLOAT), ... Show full query	▶ 2023-06-08 13:36:18 ✔ 2023-06-08 13:42:31	SUCCEEDED	fits (5.25 GiB)
14423	SELECT CAST(`kind` AS FLOAT) AS kind, CAST(`x_gal` AS FLOAT) AS x, CAST(`y_gal` AS FLOAT) AS y, CAST(`z_gal` AS ... Show full query	▶ 2023-06-08 13:29:26 ✔ 2023-06-08 13:36:37	SUCCEEDED	fits (4.37 GiB)
14407	SELECT * FROM `dmcount_edge_eli`	▶ 2023-06-07 11:59:10 ✔ 2023-06-07 12:05:20	SUCCEEDED	parquet (768.31 MiB)
14386	select reflect("java.lang.Thread", "sleep", bigint(10000000))	▶ 2023-06-01 14:40:36 ✖ 2023-06-01 14:43:57	CANCELLED	
13653	SELECT `ra_gal`, `halo_id`, `galaxy_id`, `kind`, `dec_gal`, `ra_mag_gal`, `dec_mag_gal`, `kappa`, `gamma1`, `gamma2`, ... Show full query	▶ 2023-03-29 12:56:41 ✖ 2023-03-29 13:36:35	CANCELLED	
13191	SELECT `galaxy_id`, `random_index`, `ra_gal`, `dec_gal`, `l_gal`, `b_gal`, `mw_extinction`, `hpix_29_nest`, `euclid_nisp ... Show full query	▶ 2023-01-20 16:00:36 ✔ 2023-01-20 16:09:22	DELETED	Info
13082	SELECT `x_gal`, `y_gal` FROM mice2_ia_20210203_c TABLESAMPLE (BUCKET 1 OUT OF 256)	▶ 2022-12-22 10:28:32 ✔ 2022-12-22 10:34:43	DELETED	Info
12386	SELECT `z` FROM glade_2_4_c TABLESAMPLE (BUCKET 1 OUT OF 256)	▶ 2022-09-21 10:38:59 ✔ 2022-09-21 10:44:38	DELETED	Info

UDF (I)

- HEALPix
  - Following healpy calling semantics
    - BUT using resolution order instead of NSIDE
  - Implemented functions:
    - Pixel conversion: `ang2pix, ang2vec, pix2ang, pix2vec, vec2ang, vec2pix`
    - Ordering conversion: `nest2ring, ring2nest`
    - Other: `angdist, neighbours, npix2nside, nside2npix, nside2order`

- Examples
  - Get pixel from sky coordinates
    - `SELECT udf.hp_ang2pix($ORDER, ra, dec, True) AS hpix FROM ...`
    - `SELECT udf.hp_vec2pix($ORDER, x, y, z) AS hpix FROM ...`

  - Generate PARTIAL maps
    - ```
      SELECT udf.hp_ang2pix($ORDER, ra, dec) AS hpix, AVG(redshift)
      FROM ...
      GROUP BY udf.hp_ang2pix($ORDER, ra, dec)
      ORDER BY udf.hp_ang2pix($ORDER, ra, dec)
      ```
      ```

PIC port d'informació científica

UDF (II)

- Array
  - To aggregate on equinumerous array columns
    - i.e. spectra, probability distribution functions...
  - Implemented functions:
    - `array_min, array_max, array_count, array_sum`
    - `array_avg, array_stddev_pop, array_stddev_samp, array_var_pop, array_var_samp`
- Example
  - Get average redshift probability distribution function of a sample
    - ```
      SELECT udf.array_avg(redshift_pdf) AS redshift_pdf
      FROM ...
      WHERE ...
      ```
      ```

UDF (III)

- Geometric (ADQL)
    - Deal with spherical geometries: POINT, CIRCLE, BOX/POLYGON and REGION
        - All edges are great circle arcs, including for BOX (!)
        - A REGION represents an arbitrary footprint (MOC)
        - Any geometry can be converted into a REGION
        - Performance is proportional to the number of pixels in the REGION/MOC, use moderate precision
    - Implemented functions:
        - Initialize geometries: `adql_point, adql_circle, adql_box, adql_polygon, adql_region`
        - Basic operations: `adql_area, adql_centroid, adql_coord1, adql_coord2, adql_distance`
        - Set operations: `adql_contains, adql_intersects, adql_complement`
        - Aggregations: `adql_union, adql_intersection`

UDF (III)

- Example
  - Compute approximate area of a sample
    - ```
      SELECT udf.adql_area(                    -- compute total area
          udf.adql_union(                      -- merge all regions
              udf.adql_region(                 -- convert to region
                  udf.adql_point(              -- create a point for each object
                      ra, dec
                  ), 5                         -- use a coarse resolution; NSIDE=2^5=32
              )
          )
      )
      FROM ...
      WHERE ...
      ```
      ```

Spark

```
df = spark.sql("""
  SELECT id, ra, dec
  FROM cosmohub.micecatv1_0_hpix
  LIMIT 100
""")
df
```

```
DataFrame[id: int, ra: double, dec: double]
```

```
df.show(5)
```

```
+---------+---------+---------+
|       id|       ra|      dec|
+---------+---------+---------+
|191225057|18.523232|79.887398|
| 49810401|59.949303|20.816753|
|  9887201| 22.78075|46.971172|
| 11503841|51.193577| 17.27203|
| 43089377| 8.418952|16.733221|
+---------+---------+---------+
only showing top 5 rows
```

```
df.rdd.take(5)
```

```
[Row(id=69631390, ra=18.716858, dec=9.833587),
 Row(id=152771998, ra=58.511005, dec=24.961664),
 Row(id=154729374, ra=35.783665, dec=8.607564),
 Row(id=130099870, ra=79.352654, dec=8.510796),
 Row(id=99296158, ra=89.297478, dec=0.731858)]
```

PIC
port d'informació
científica

Spark

```
# Convert to Pandas chunks
from scipic.mocks import spark_utils

df_chunks = df.rdd.mapPartitionsWithIndex(
    spark_utils.to_df_chunks(
        chunk_size = 5,
        index_cols = ['id'],
        dtypes = df.dtypes,
    )
)
```

```
df_chunks.take(1)[0]
```

	ra	dec
id		
102285412	65.593170	51.391582
25057892	25.937380	7.770375
84641380	18.656963	11.537212
106692452	29.782038	80.949181
45668449	11.779725	23.124931

```
df_chunks.count()
```

20

PIC port d'informació científica

Spark

```python
# Define pipeline
import healpy as hp

def my_pipeline(df):
    df['hpix'] = hp.ang2pix(32, df['ra'], df['dec'], lonlat=True)
    return df
```

```python
df_result = df_chunks.map(
    my_pipeline
)
```

```python
df_result.take(1)[0]
```

id	ra	dec	hpix
102285412	65.593170	51.391582	1318
25057892	25.937380	7.770375	5193
84641380	18.656963	11.537212	4806
106692452	29.782038	80.949181	86
45668449	11.779725	23.124931	3652

PIC port d'informació científica

Spark

```
# Store as Parquet into HDFS
count = df_result.mapPartitionsWithIndex(
    spark_utils.to_parquet('/user/tallada/data/euclid/test_result/{}.pq')
).sum()

# Map the Parquet files to a table i my own schema
spark.catalog.createTable('tallada.test_result', path='/user/tallada/data/euclid/test_result')

print(f"{count} objects stored.")
```

```
100 objects stored.
```

```
df = spark.sql("""
  SELECT *
  FROM tallada.test_result
""")
df
```

```
DataFrame[ra: double, dec: double, hpix: bigint, id: bigint]
```

```
df.show(5)
```

```
+---------+---------+----+---------+
|       ra|      dec|hpix|       id|
+---------+---------+----+---------+
| 65.59317|51.391582|1318|102285412|
| 25.93738| 7.770375|5193| 25057892|
|18.656963|11.537212|4806| 84641380|
|29.782038|80.949181|  86|106692452|
|11.779725|23.124931|3652| 45668449|
+---------+---------+----+---------+
only showing top 5 rows
```