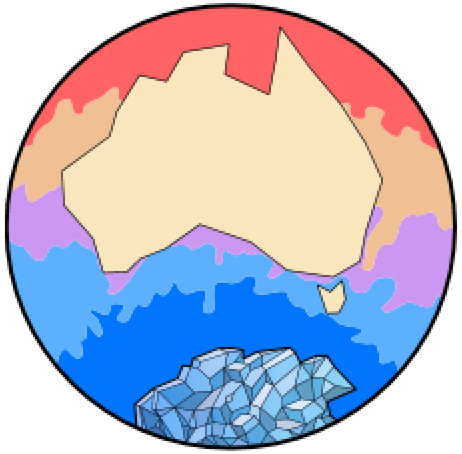


Building your own intake-esm datastore

Standardising analysis of climate model datasets

Anton Steketee, Charles Turner, Dougie Squire, Marc White, Romain Beucher

ACCESS-NRI Intake Catalog

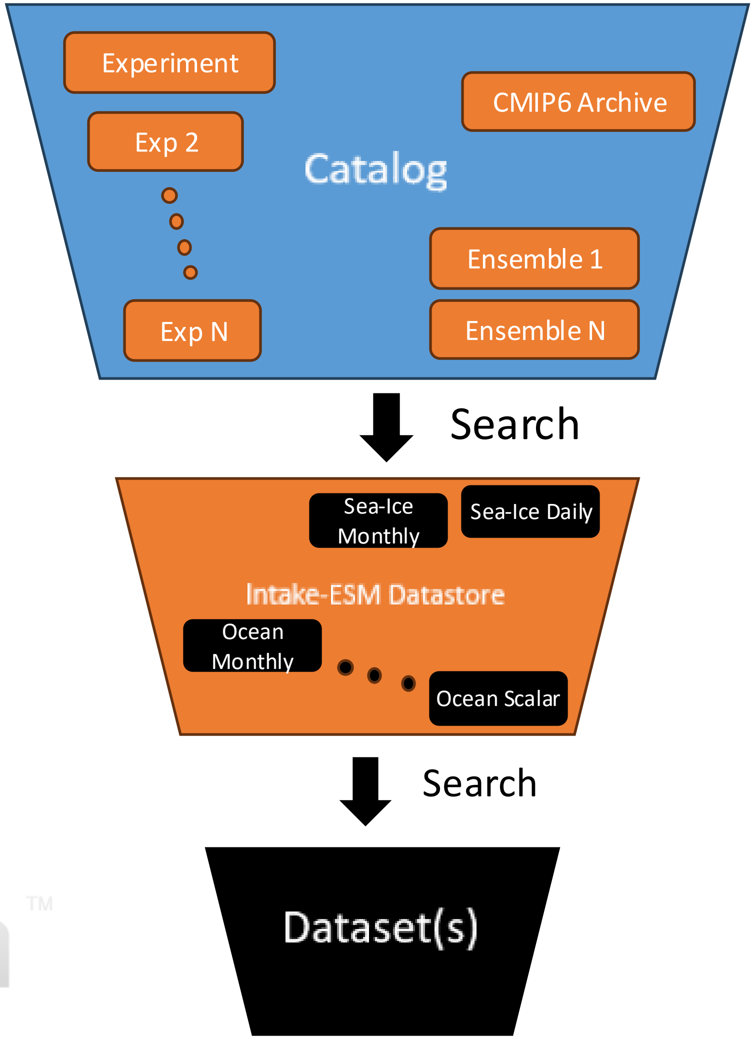
A catalog provides functionality for searching, discovering and loading data.

Data can be searched by many attributes, including:

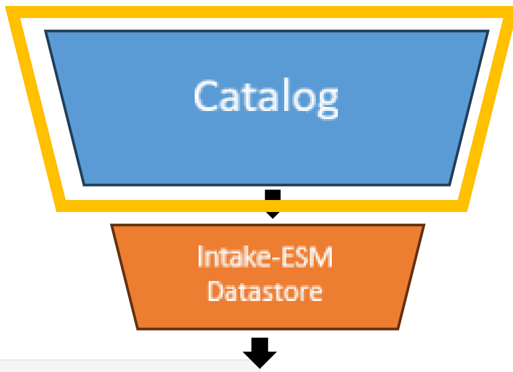
- Experiment name
- Model
- Realm / Model Component
- Data Frequency
- Variable Name
- Variable Standard & Long Names

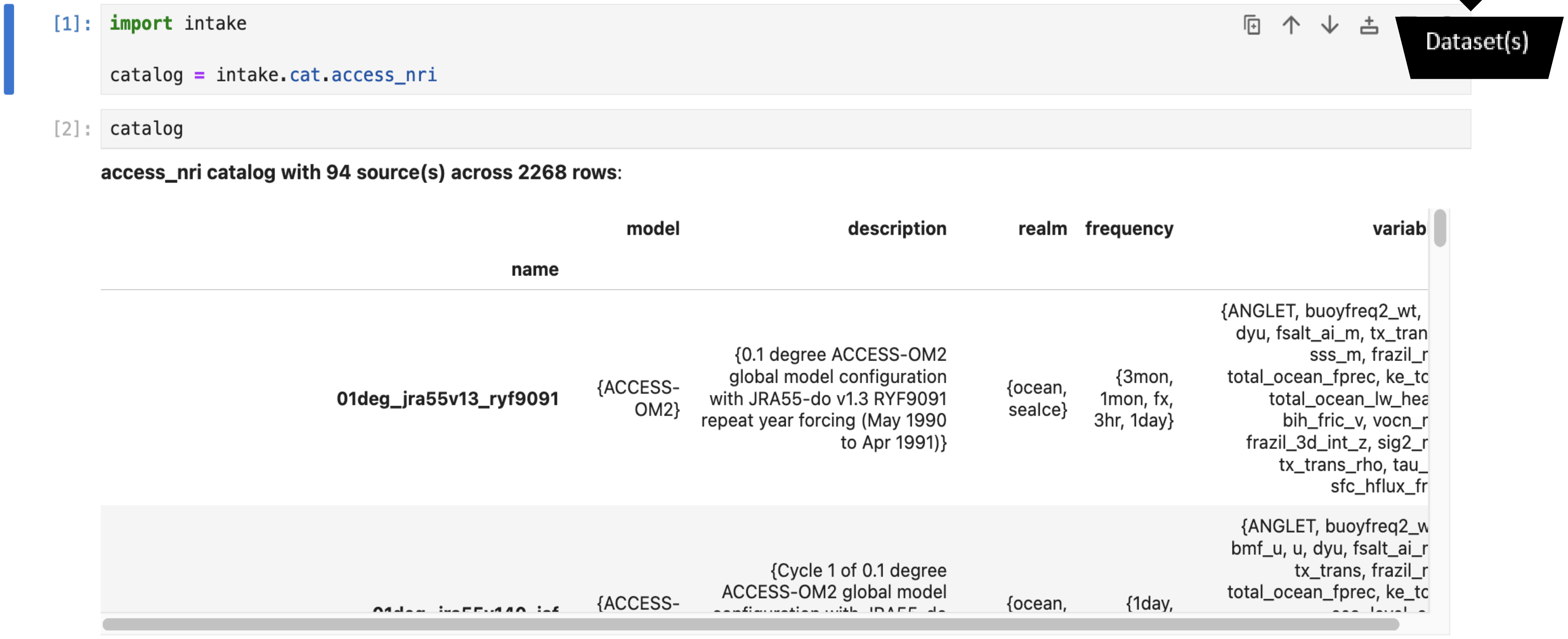
The ACCESS-NRI Intake Catalog is built upon Intake-ESM, and only shows data stored at the NCI.

xarray jupyter python™



ACCESS-NRI Intake Catalog





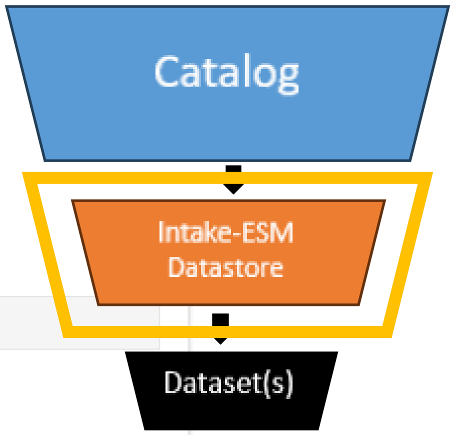
```
[1]: import intake

     catalog = intake.cat.access_nri
```

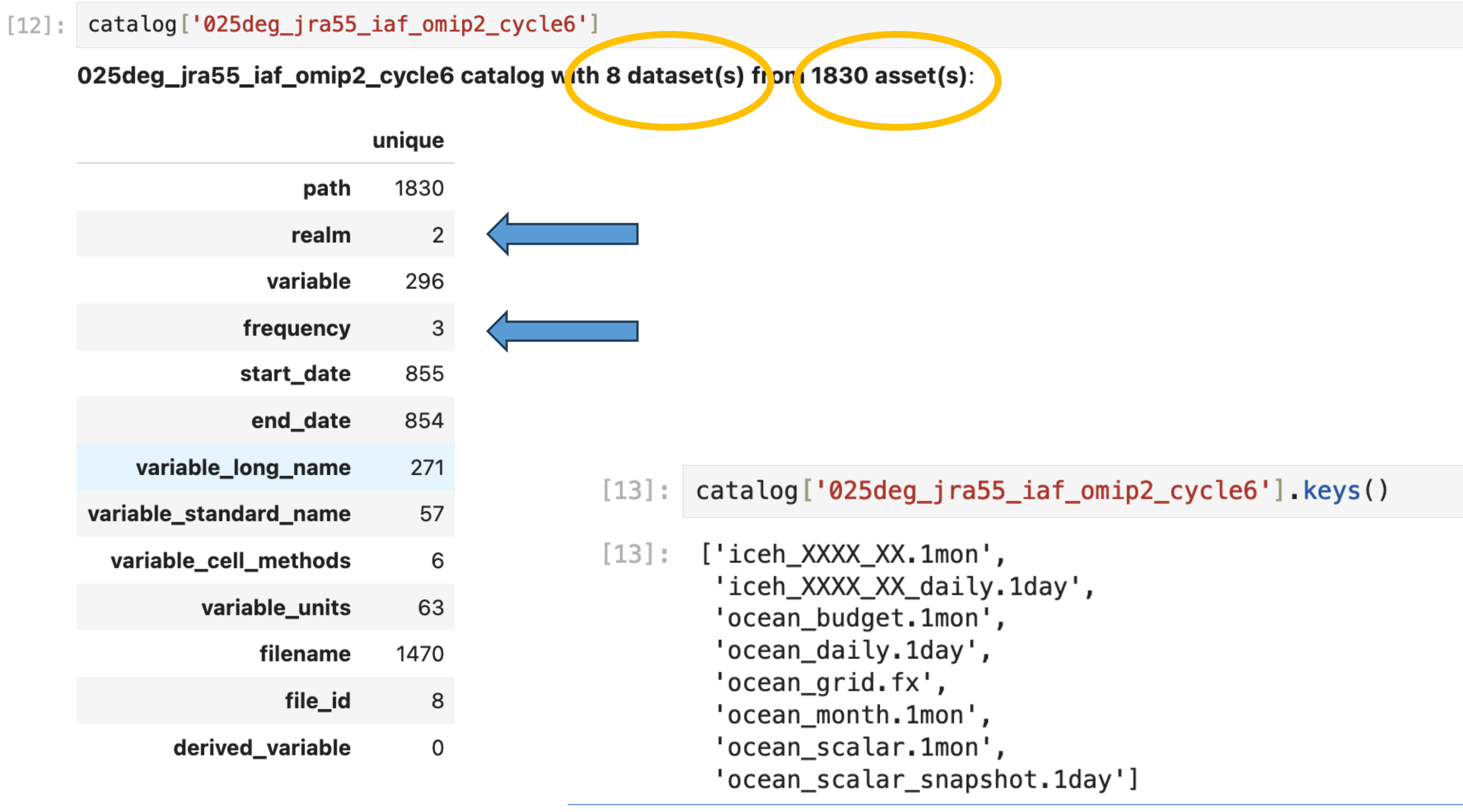
```
[2]: catalog
```

access_nri catalog with 94 source(s) across 2268 rows:

name	model	description	realm	frequency	variab
01deg_jra55v13_ryf9091	{ACCESS-OM2}	{0.1 degree ACCESS-OM2 global model configuration with JRA55-do v1.3 RYF9091 repeat year forcing (May 1990 to Apr 1991)}	{ocean, sealce}	{3mon, 1mon, fx, 3hr, 1day}	{ANGLET, buoyfreq2_wt, dyu, fsalt_ai_m, tx_tran sss_m, frazil_r total_ocean_fprec, ke_tc total_ocean_lw_hea bih_fric_v, vocn_r frazil_3d_int_z, sig2_r tx_trans_rho, tau_ sfc_hflux_fr
01deg_jra55v140_iaf	{ACCESS-	{Cycle 1 of 0.1 degree ACCESS-OM2 global model configuration with JRA55-do	{ocean,	{1day,	{ANGLET, buoyfreq2_w bmf_u, u, dyu, fsalt_ai_r tx_trans, frazil_r total_ocean_fprec, ke_tc

Each experiment is an Intake-ESM Datastore





```
[12]: catalog['025deg_jra55_iaf_omip2_cycle6']
```

025deg_jra55_iaf_omip2_cycle6 catalog with 8 dataset(s) from 1830 asset(s):

	unique
path	1830
realm	2
variable	296
frequency	3
start_date	855
end_date	854
variable_long_name	271
variable_standard_name	57
variable_cell_methods	6
variable_units	63
filename	1470
file_id	8
derived_variable	0

```
[13]: catalog['025deg_jra55_iaf_omip2_cycle6'].keys()

[13]: ['iceh_XXXX_XX.1mon',
       'iceh_XXXX_XX_daily.1day',
       'ocean_budget.1mon',
       'ocean_daily.1day',
       'ocean_grid.fx',
       'ocean_month.1mon',
       'ocean_scalar.1mon',
       'ocean_scalar_snapshot.1day']
```

How do I make my own Intake ESM Datastore?

Example with build-esm-datastore in command line

Through the command line: `build-esm-datastore` :

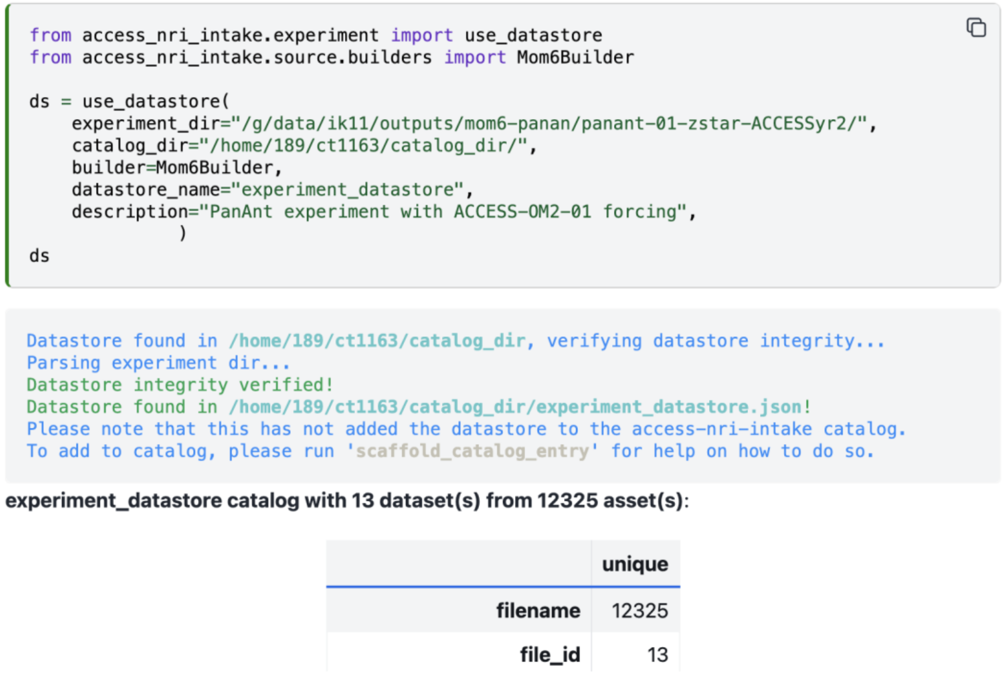
```
user@gadi $ mkdir catalog_dir  # Change this to whatever you like/seems sensible to you
user@gadi $ module load conda/analysis3-25.02
user@gadi $ build-esm-datastore --builder Mom6Builder --expt-dir
 /g/data/ik11/outputs/mom6-panan/panant-01-zstar-ACCESSyr2/ --cat-dir catalog_dir
```

In a Jupyter Notebook: use_datastore:

```
from access_nri_intake.experiment import use_datastore
from access_nri_intake.source import builders

expt_datastore = use_datastore(
    experiment_dir = '/g/data/ik11/outputs/mom6-panan/panant-01-zstar-ACCESSyr2/',
    catalog_dir = 'catalog_dir',
    builder=builders.Mom6Builder
)
```

How do I make my own Intake ESM Datastore?

Further info: https://access-nri-intake-catalog.readthedocs.io/en/latest/datastores/quickstart.html

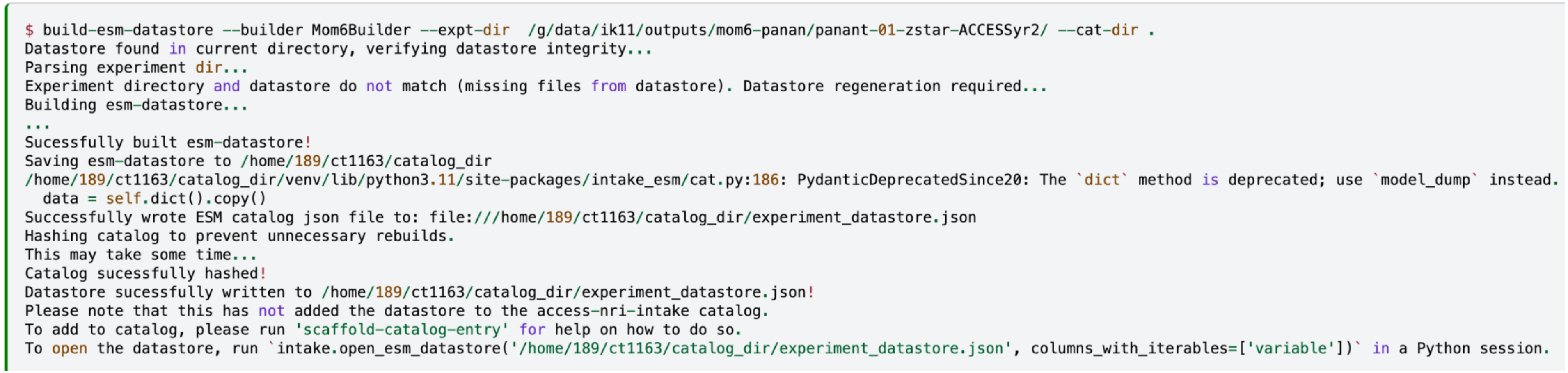


```
from access_nri_intake.experiment import use_datastore
from access_nri_intake.source.builders import Mom6Builder

ds = use_datastore(
    experiment_dir="/g/data/ik11/outputs/mom6-panan/panant-01-zstar-ACCESSyr2/",
    catalog_dir="/home/189/ct1163/catalog_dir/",
    builder=Mom6Builder,
    datastore_name="experiment_datastore",
    description="PanAnt experiment with ACCESS-OM2-01 forcing",
              )
ds
```

```
Datastore found in /home/189/ct1163/catalog_dir, verifying datastore integrity...
Parsing experiment dir...
Datastore integrity verified!
Datastore found in /home/189/ct1163/catalog_dir/experiment_datastore.json!
Please note that this has not added the datastore to the access-nri-intake catalog.
To add to catalog, please run 'scaffold_catalog_entry' for help on how to do so.
```

experiment_datastore catalog with 13 dataset(s) from 12325 asset(s):

	unique
filename	12325
file_id	13

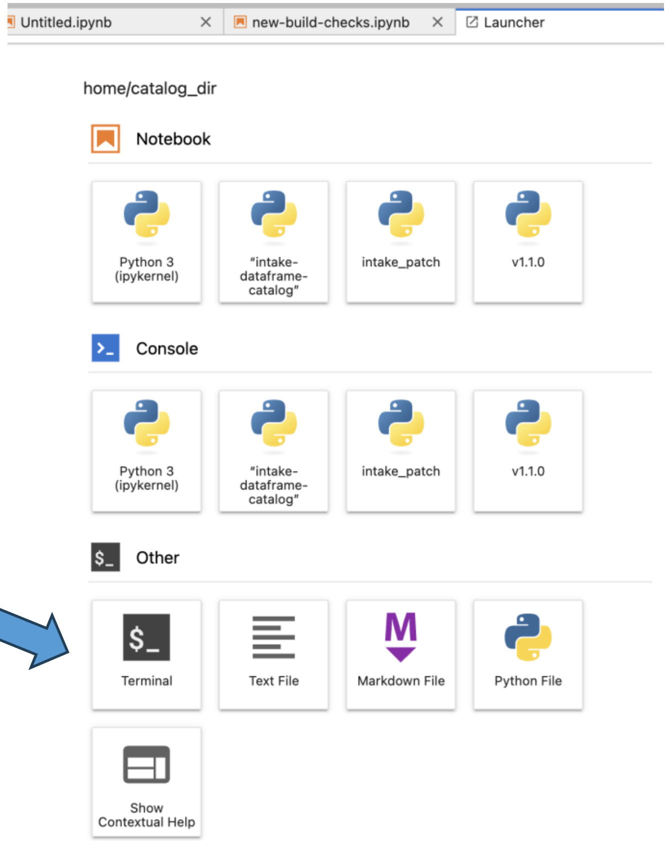


```
$ build-esm-datastore --builder Mom6Builder --expt-dir  /g/data/ik11/outputs/mom6-panan/panant-01-zstar-ACCESSyr2/ --cat-dir .
Datastore found in current directory, verifying datastore integrity...
Parsing experiment dir...
Experiment directory and datastore do not match (missing files from datastore). Datastore regeneration required...
Building esm-datastore...
...
Sucessfully built esm-datastore!
Saving esm-datastore to /home/189/ct1163/catalog_dir
/home/189/ct1163/catalog_dir/venv/lib/python3.11/site-packages/intake_esm/cat.py:186: PydanticDeprecatedSince20: The `dict` method is deprecated; use `model_dump` instead.
  data = self.dict().copy()
Successfully wrote ESM catalog json file to: file:///home/189/ct1163/catalog_dir/experiment_datastore.json
Hashing catalog to prevent unnecessary rebuilds.
This may take some time...
Catalog sucessfully hashed!
Datastore sucessfully written to /home/189/ct1163/catalog_dir/experiment_datastore.json!
Please note that this has not added the datastore to the access-nri-intake catalog.
To add to catalog, please run 'scaffold-catalog-entry' for help on how to do so.
To open the datastore, run `intake.open_esm_datastore('/home/189/ct1163/catalog_dir/experiment_datastore.json', columns_with_iterables=['variable'])` in a Python session.
```

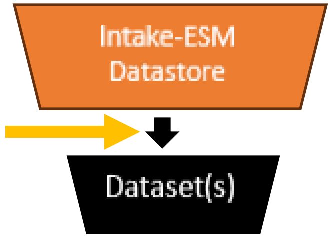
How do I make my own Intake ESM Datastore?

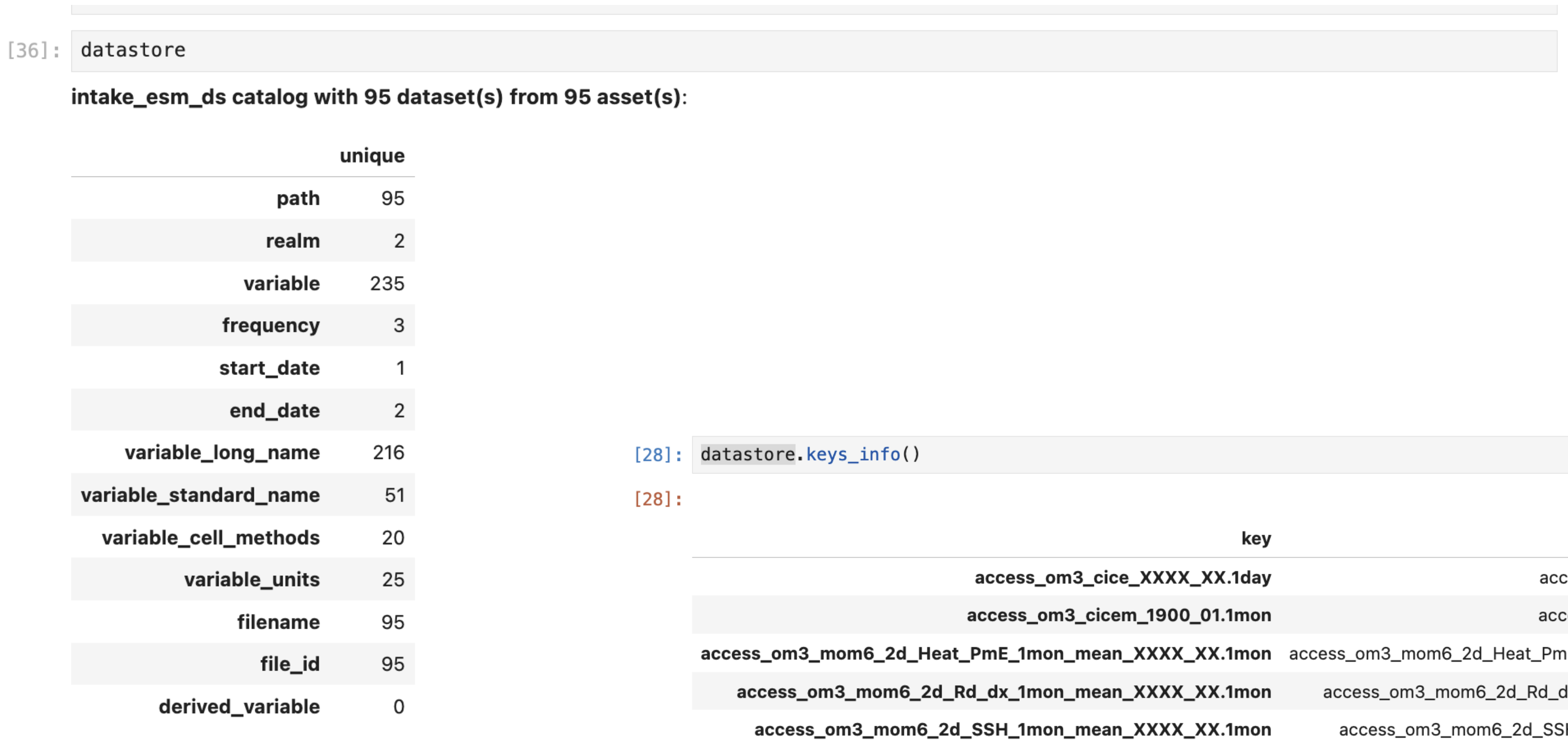
Further info: https://access-nri-intake-catalog.readthedocs.io/en/latest/datastores/quickstart.html

If building a catalog from the terminal, we recommend you do so from an ARE instance terminal, not by SSH'ing into a Gadi login node



Datastore keys define different datasets

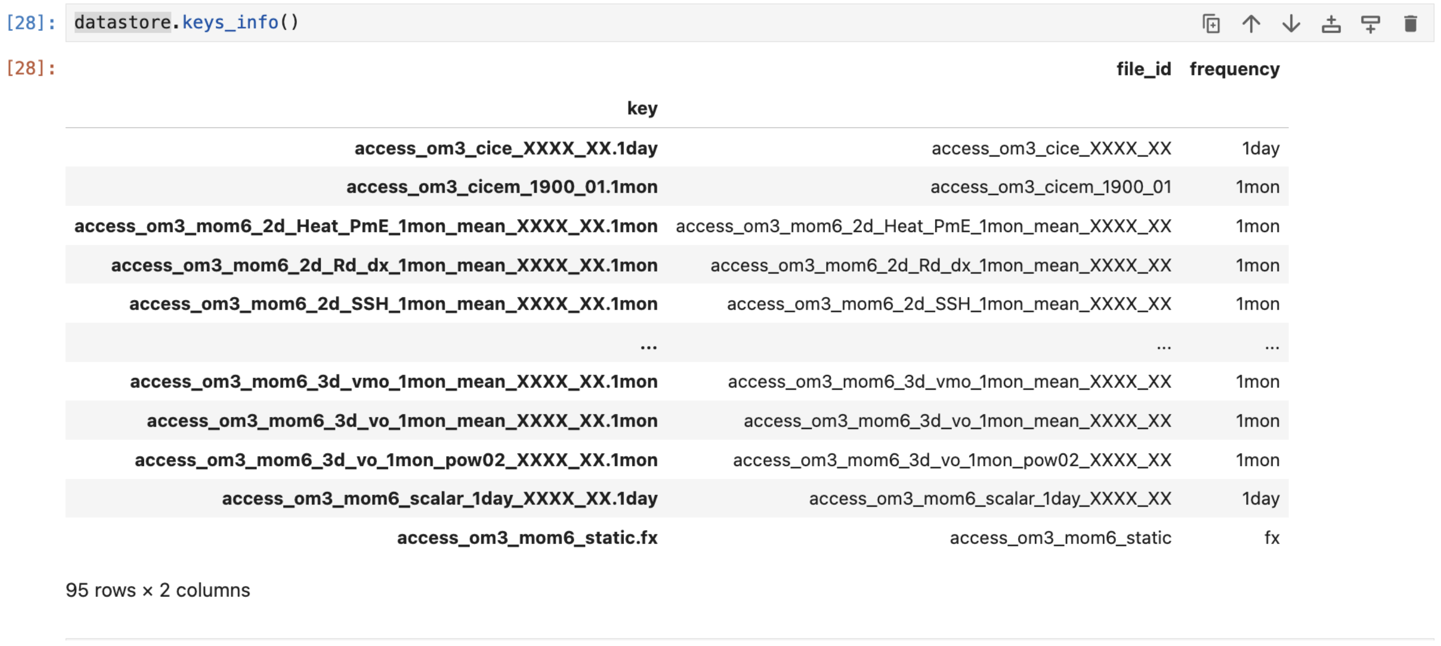




```
[36]: datastore
```

intake_esm_ds catalog with 95 dataset(s) from 95 asset(s):

	unique
path	95
realm	2
variable	235
frequency	3
start_date	1
end_date	2
variable_long_name	216
variable_standard_name	51
variable_cell_methods	20
variable_units	25
filename	95
file_id	95
derived_variable	0



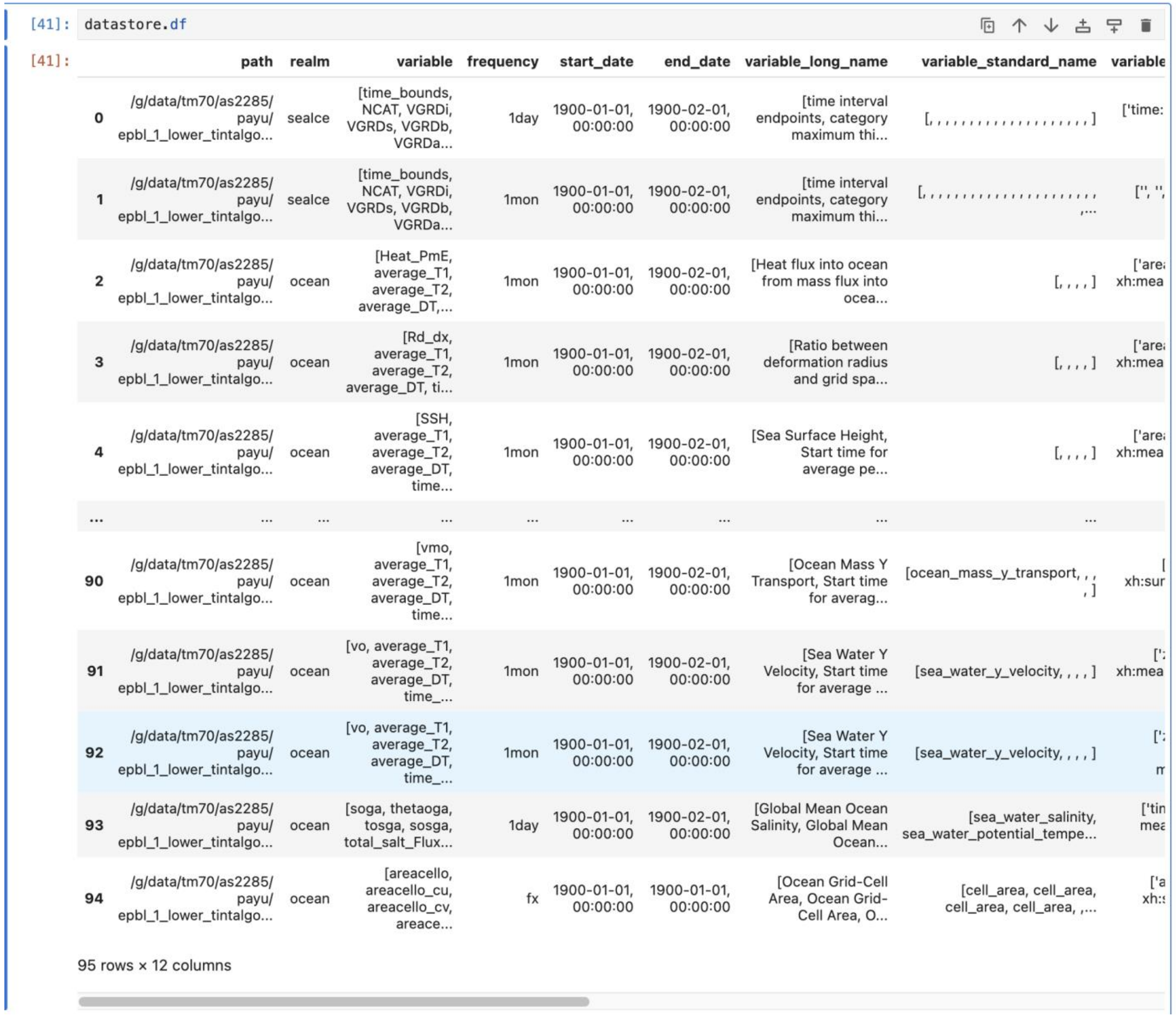
```
[28]: datastore.keys_info()
```

[28]:

key	file_id	frequency
access_om3_cice_XXXX_XX.1day	access_om3_cice_XXXX_XX	1day
access_om3_cicem_1900_01.1mon	access_om3_cicem_1900_01	1mon
access_om3_mom6_2d_Heat_PmE_1mon_mean_XXXX_XX.1mon	access_om3_mom6_2d_Heat_PmE_1mon_mean_XXXX_XX	1mon
access_om3_mom6_2d_Rd_dx_1mon_mean_XXXX_XX.1mon	access_om3_mom6_2d_Rd_dx_1mon_mean_XXXX_XX	1mon
access_om3_mom6_2d_SSH_1mon_mean_XXXX_XX.1mon	access_om3_mom6_2d_SSH_1mon_mean_XXXX_XX	1mon
...	...	...
access_om3_mom6_3d_vmo_1mon_mean_XXXX_XX.1mon	access_om3_mom6_3d_vmo_1mon_mean_XXXX_XX	1mon
access_om3_mom6_3d_vo_1mon_mean_XXXX_XX.1mon	access_om3_mom6_3d_vo_1mon_mean_XXXX_XX	1mon
access_om3_mom6_3d_vo_1mon_pow02_XXXX_XX.1mon	access_om3_mom6_3d_vo_1mon_pow02_XXXX_XX	1mon
access_om3_mom6_scalar_1day_XXXX_XX.1day	access_om3_mom6_scalar_1day_XXXX_XX	1day
access_om3_mom6_static.fx	access_om3_mom6_static	fx

95 rows × 2 columns

Dataframe view shows assets



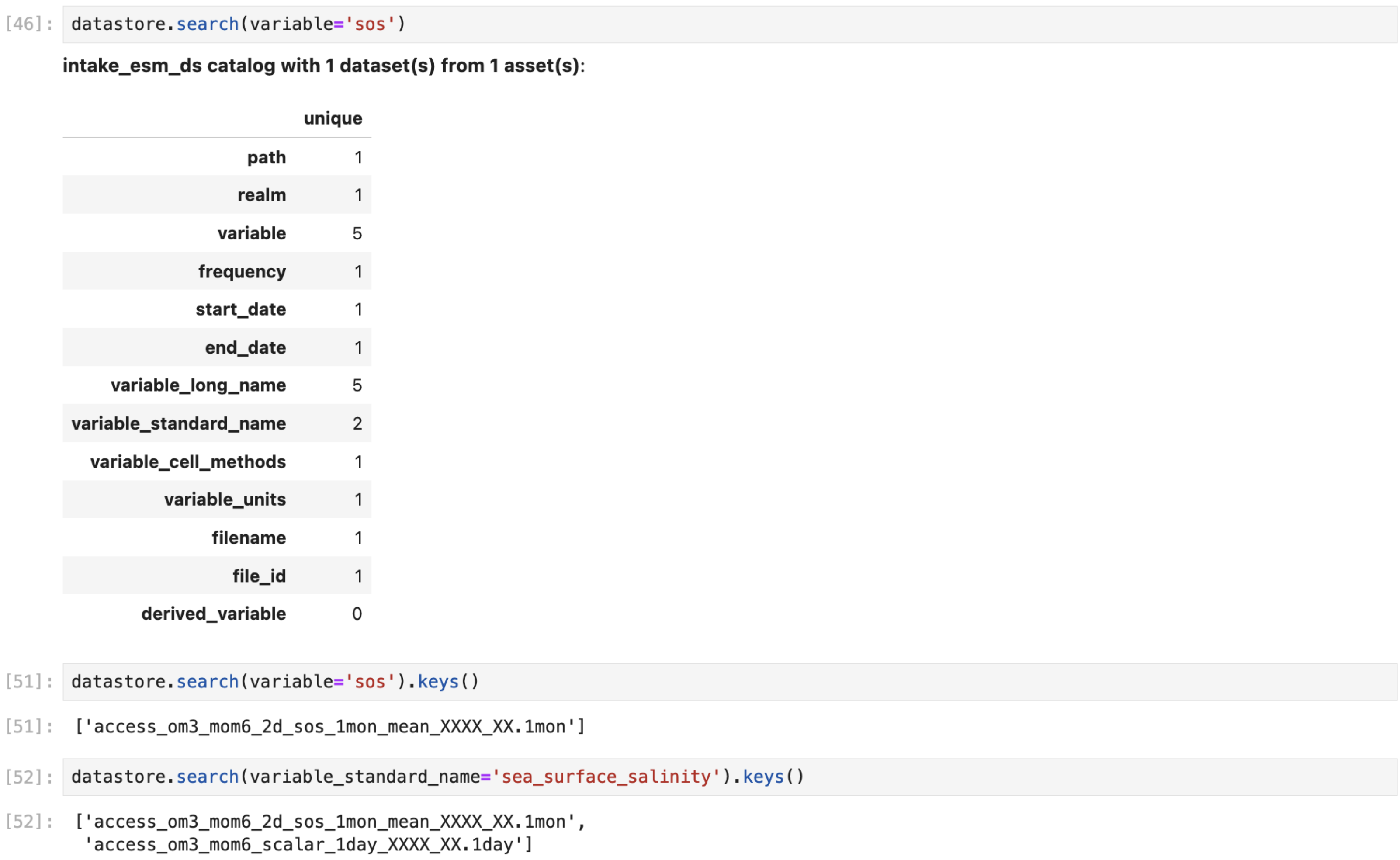
```
[41]: datastore.df
```

[41]:

	path	realm	variable	frequency	start_date	end_date	variable_long_name	variable_standard_name	variable
0	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	seaIce	[time_bounds, NCAT, VGRDi, VGRDs, VGRDb, VGRDa...	1day	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[time interval endpoints, category maximum thi...	[,]	['time:
1	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	seaIce	[time_bounds, NCAT, VGRDi, VGRDs, VGRDb, VGRDa...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[time interval endpoints, category maximum thi...	[, ,...	['', '',
2	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[Heat_PmE, average_T1, average_T2, average_DT,...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Heat flux into ocean from mass flux into ocea...	[, , , ,]	['area xh:mea
3	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[Rd_dx, average_T1, average_T2, average_DT, ti...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Ratio between deformation radius and grid spa...	[, , , ,]	['area xh:mea
4	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[SSH, average_T1, average_T2, average_DT, time...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Sea Surface Height, Start time for average pe...	[, , , ,]	['area xh:mea
...	...	...	...	...	...	...	...	...	
90	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[vmo, average_T1, average_T2, average_DT, time...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Ocean Mass Y Transport, Start time for averag...	[ocean_mass_y_transport, , , ,]	[xh:sur
91	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[vo, average_T1, average_T2, average_DT, time_...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Sea Water Y Velocity, Start time for average ...	[sea_water_y_velocity, , , ,]	[': xh:mea
92	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[vo, average_T1, average_T2, average_DT, time_...	1mon	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Sea Water Y Velocity, Start time for average ...	[sea_water_y_velocity, , , ,]	[': m
93	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[soga, thetaoga, tosga, sosga, total_salt_Flux...	1day	1900-01-01, 00:00:00	1900-02-01, 00:00:00	[Global Mean Ocean Salinity, Global Mean Ocean...	[sea_water_salinity, sea_water_potential_tempe...	['tin mea
94	/g/data/tm70/as2285/ payu/ epbl_1_lower_tintalgo...	ocean	[areacello, areacello_cu, areacello_cv, areace...	fx	1900-01-01, 00:00:00	1900-01-01, 00:00:00	[Ocean Grid-Cell Area, Ocean Grid-Cell Area, O...	[cell_area, cell_area, cell_area, cell_area, ,...	['a xh:

95 rows × 12 columns

Finding a variable by name

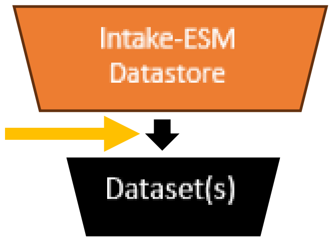


```
[46]: datastore.search(variable='sos')
```

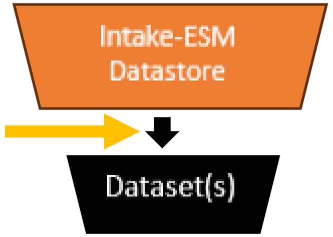
intake_esm_ds catalog with 1 dataset(s) from 1 asset(s):

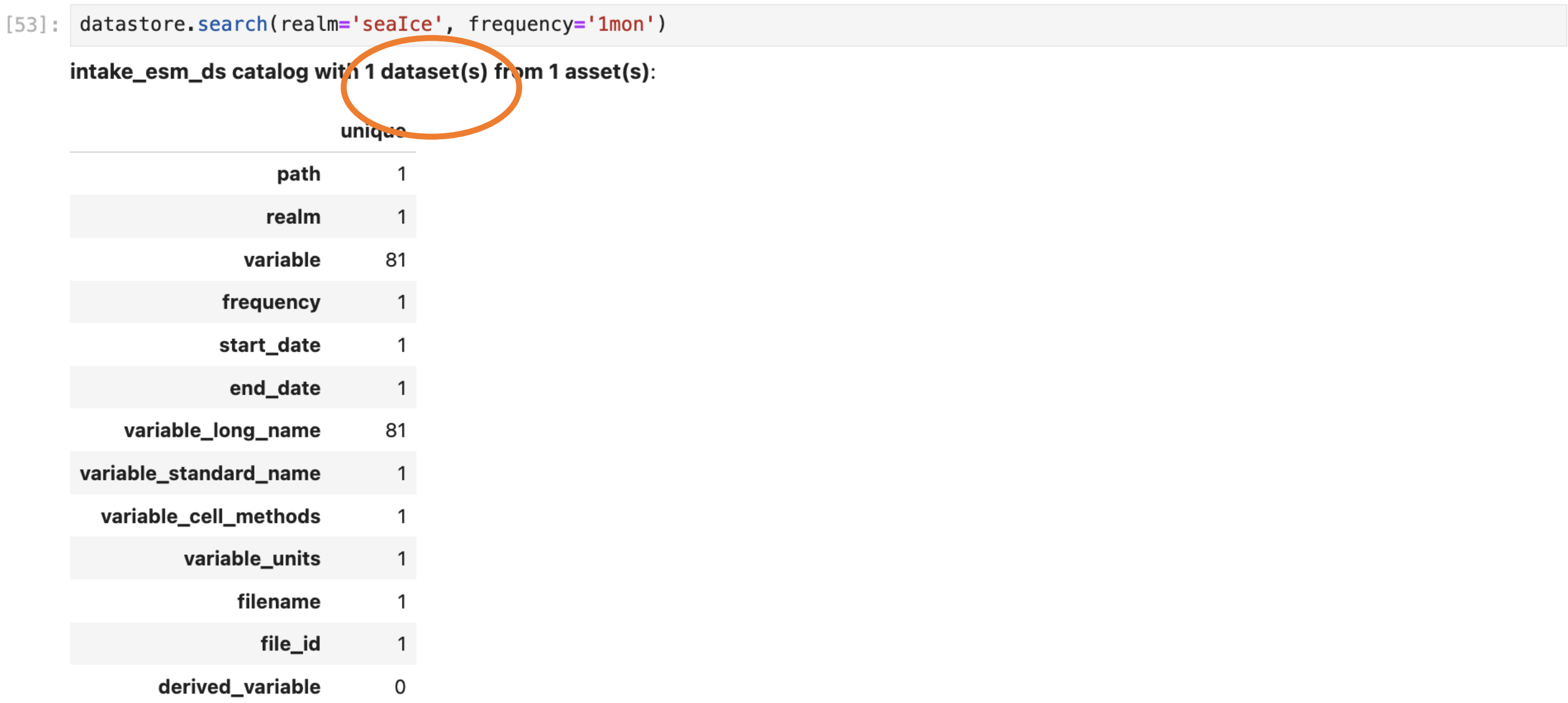
	unique
path	1
realm	1
variable	5
frequency	1
start_date	1
end_date	1
variable_long_name	5
variable_standard_name	2
variable_cell_methods	1
variable_units	1
filename	1
file_id	1
derived_variable	0

```
[51]: datastore.search(variable='sos').keys()

[51]: ['access_om3_mom6_2d_sos_1mon_mean_XXXX_XX.1mon']

[52]: datastore.search(variable_standard_name='sea_surface_salinity').keys()

[52]: ['access_om3_mom6_2d_sos_1mon_mean_XXXX_XX.1mon',
       'access_om3_mom6_scalar_1day_XXXX_XX.1day']
```



Finding a dataset by attribute



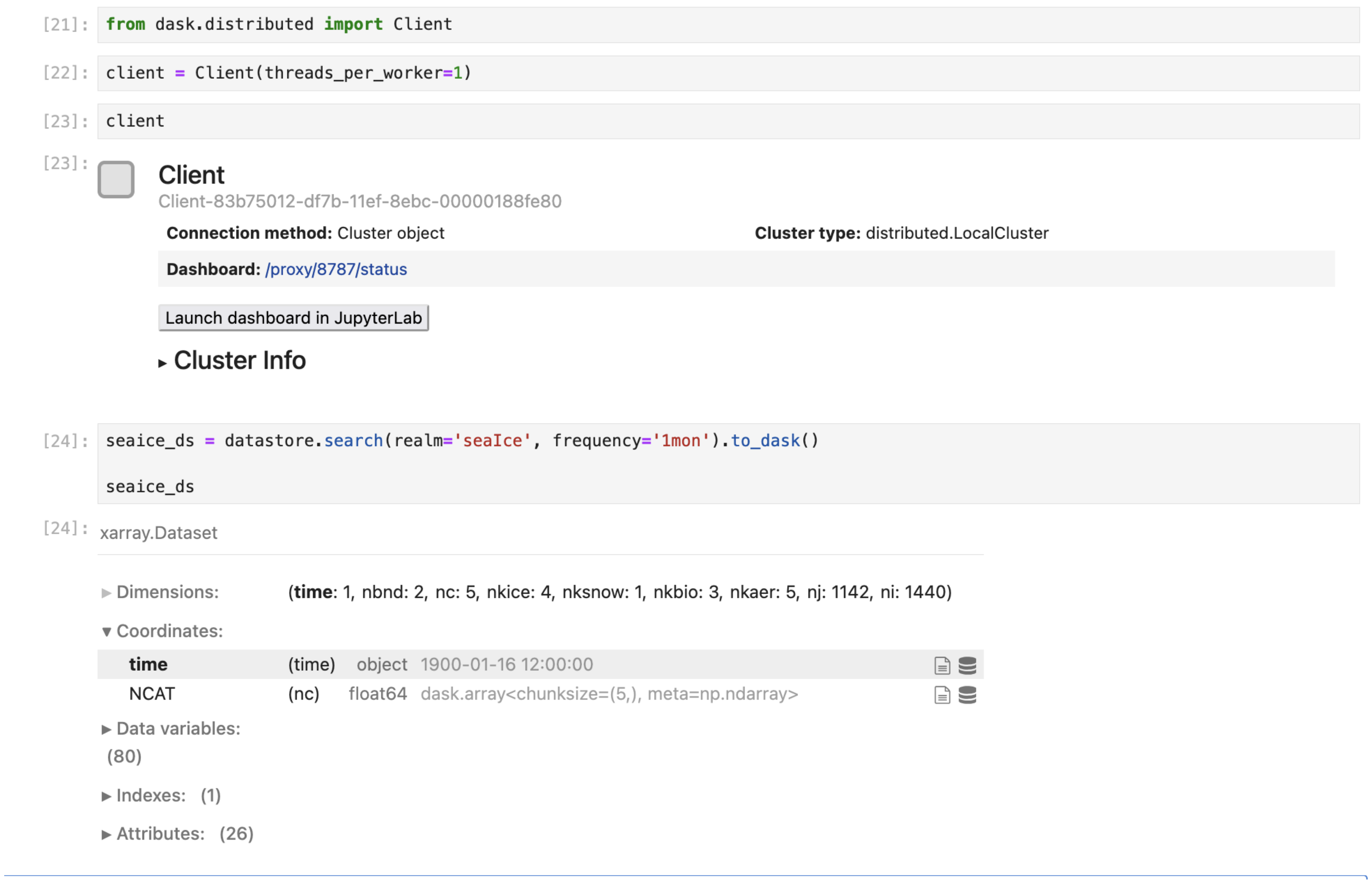


```
[53]: datastore.search(realm='seaIce', frequency='1mon')
```

intake_esm_ds catalog with 1 dataset(s) from 1 asset(s):

	unique
path	1
realm	1
variable	81
frequency	1
start_date	1
end_date	1
variable_long_name	81
variable_standard_name	1
variable_cell_methods	1
variable_units	1
filename	1
file_id	1
derived_variable	0

Opening a dataset with multiple variables

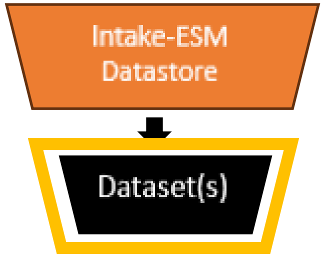


```
[21]: from dask.distributed import Client

[22]: client = Client(threads_per_worker=1)

[23]: client
```

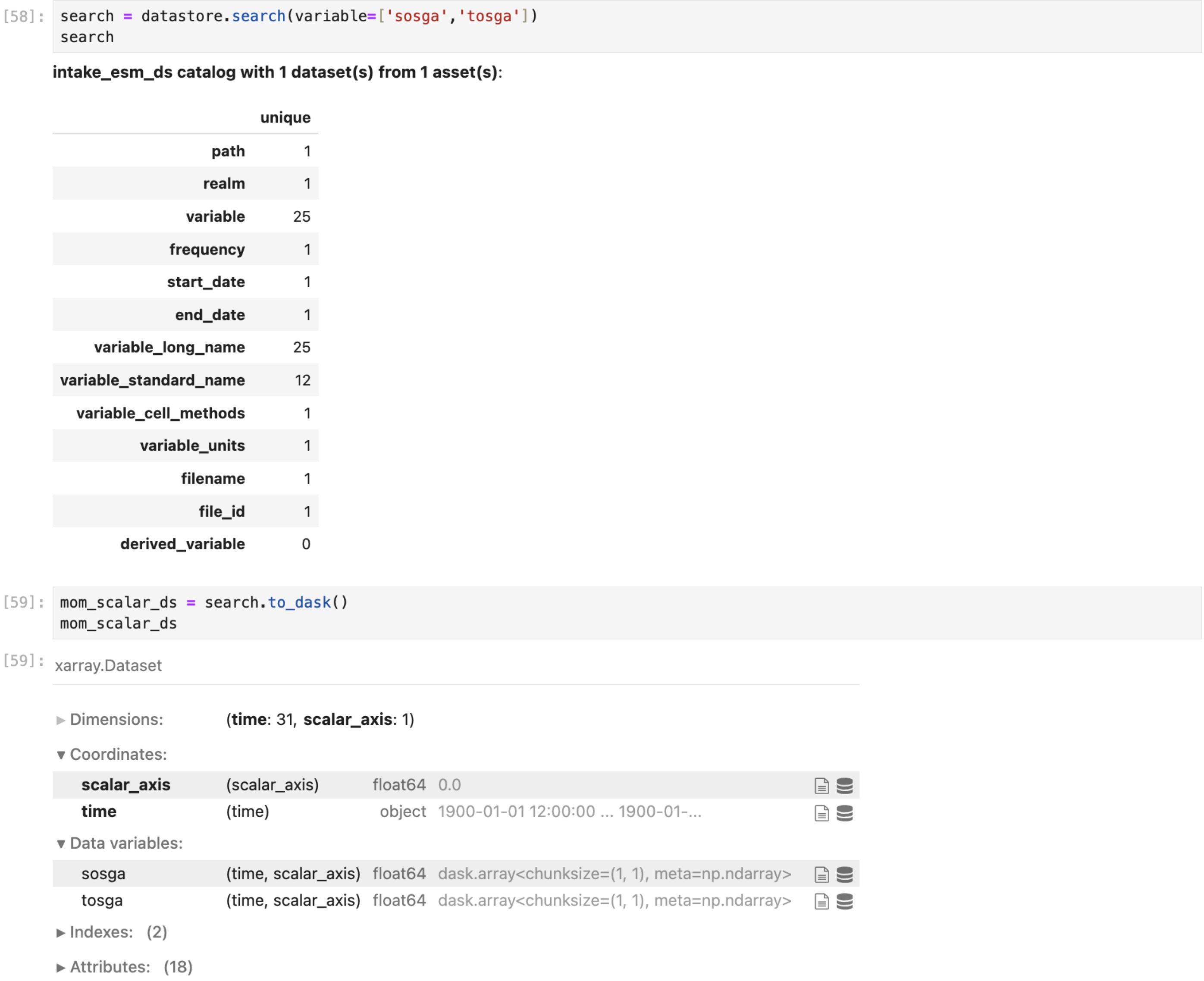
```
[23]:
Client
Client-83b75012-df7b-11ef-8ebc-00000188fe80
Connection method: Cluster object                Cluster type: distributed.LocalCluster
Dashboard: /proxy/8787/status

Launch dashboard in JupyterLab

▸ Cluster Info
```

```
[24]: seaice_ds = datastore.search(realm='seaIce', frequency='1mon').to_dask()

      seaice_ds
```

```
[24]: xarray.Dataset

▸ Dimensions:       (time: 1, nbnd: 2, nc: 5, nkice: 4, nksnow: 1, nkbio: 3, nkaer: 5, nj: 1142, ni: 1440)
▾ Coordinates:
    time            (time)   object   1900-01-16 12:00:00
    NCAT            (nc)     float64  dask.array<chunksize=(5,), meta=np.ndarray>
▸ Data variables:
  (80)
▸ Indexes:  (1)
▸ Attributes:  (26)
```



Datasets can have more than one variable



```
[58]: search = datastore.search(variable=['sosga','tosga'])
      search
```

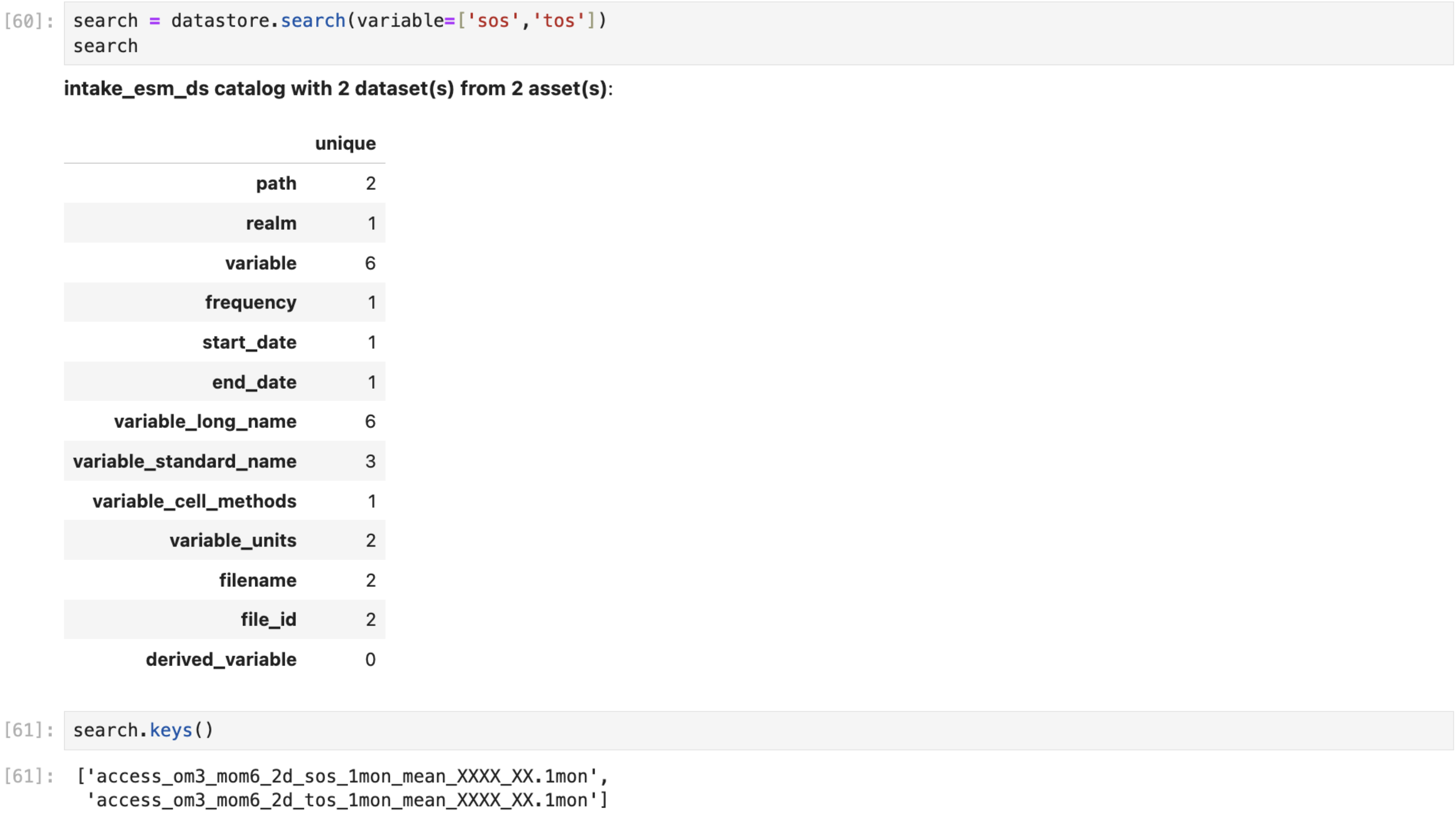
intake_esm_ds catalog with 1 dataset(s) from 1 asset(s):

	unique
path	1
realm	1
variable	25
frequency	1
start_date	1
end_date	1
variable_long_name	25
variable_standard_name	12
variable_cell_methods	1
variable_units	1
filename	1
file_id	1
derived_variable	0

```
[59]: mom_scalar_ds = search.to_dask()
      mom_scalar_ds
```

```
[59]: xarray.Dataset

▶ Dimensions:      (time: 31, scalar_axis: 1)
▼ Coordinates:
    scalar_axis    (scalar_axis)        float64  0.0
    time           (time)               object   1900-01-01 12:00:00 ... 1900-01-...
▼ Data variables:
    sosga          (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np.ndarray>
    tosga          (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np.ndarray>
▶ Indexes:   (2)
▶ Attributes:   (18)
```

Related variables are often in different datasets



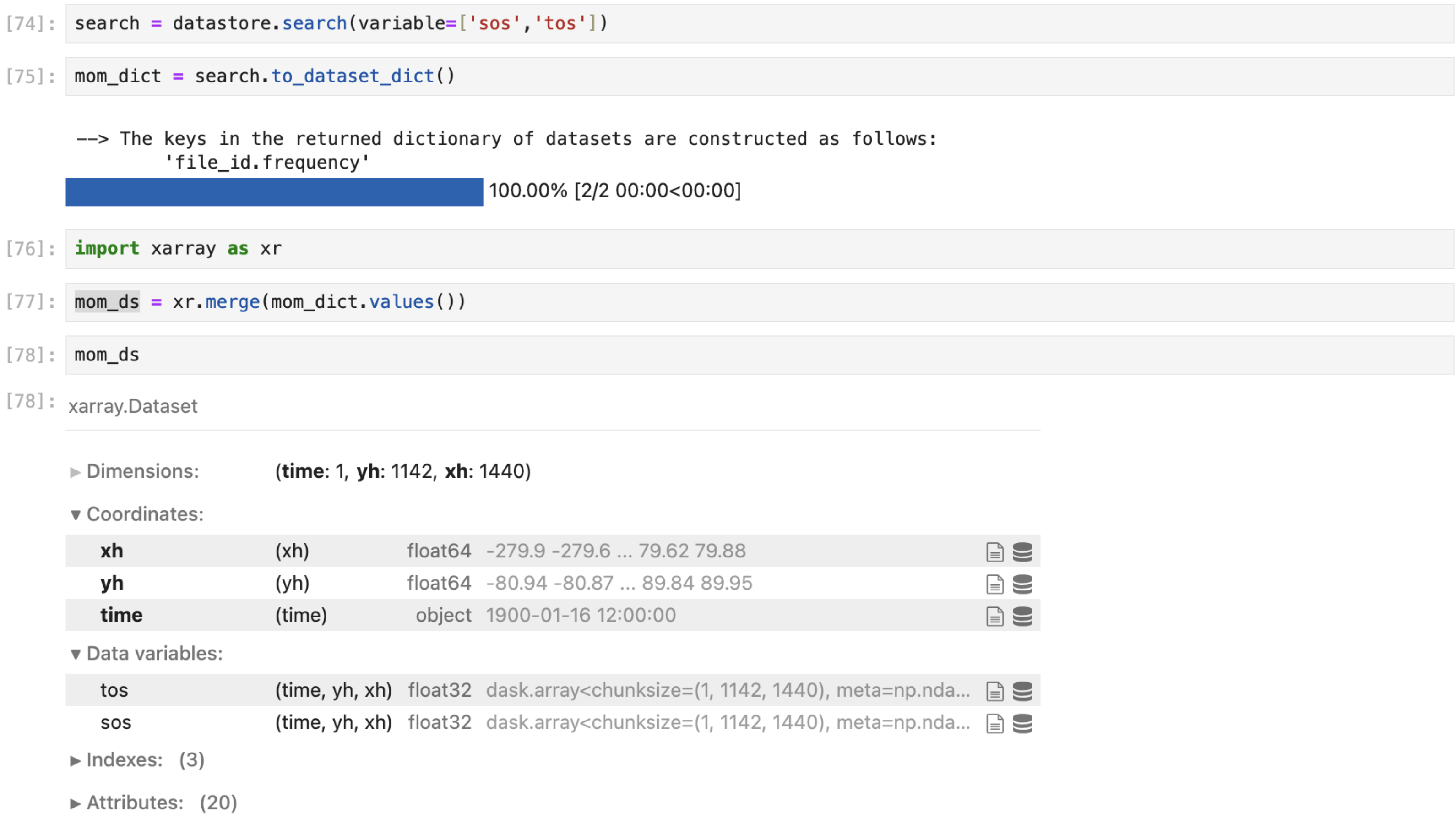
```
[60]: search = datastore.search(variable=['sos','tos'])
      search
```

intake_esm_ds catalog with 2 dataset(s) from 2 asset(s):

	unique
path	2
realm	1
variable	6
frequency	1
start_date	1
end_date	1
variable_long_name	6
variable_standard_name	3
variable_cell_methods	1
variable_units	2
filename	2
file_id	2
derived_variable	0

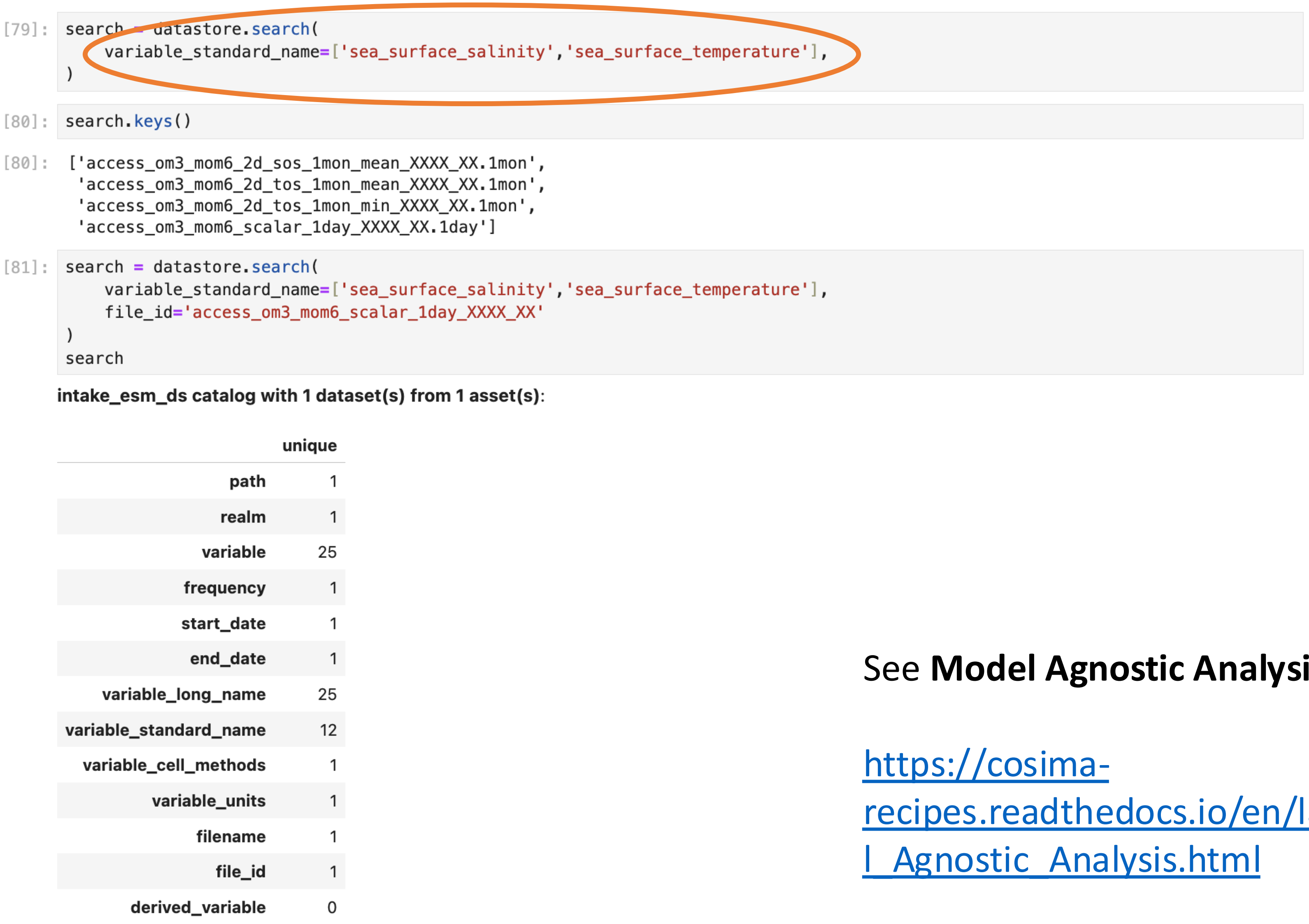
```
[61]: search.keys()

[61]: ['access_om3_mom6_2d_sos_1mon_mean_XXXX_XX.1mon',
       'access_om3_mom6_2d_tos_1mon_mean_XXXX_XX.1mon']
```

Related variables on the same grid can be merged



```
[74]: search = datastore.search(variable=['sos','tos'])

[75]: mom_dict = search.to_dataset_dict()

      --> The keys in the returned dictionary of datasets are constructed as follows:
              'file_id.frequency'
      100.00% [2/2 00:00<00:00]

[76]: import xarray as xr

[77]: mom_ds = xr.merge(mom_dict.values())

[78]: mom_ds

[78]: xarray.Dataset

      ▸ Dimensions:     (time: 1, yh: 1142, xh: 1440)
      ▾ Coordinates:
          xh            (xh)             float64   -279.9 -279.6 ... 79.62 79.88
          yh            (yh)             float64   -80.94 -80.87 ... 89.84 89.95
          time          (time)            object   1900-01-16 12:00:00
      ▾ Data variables:
          tos           (time, yh, xh)   float32   dask.array<chunksize=(1, 1142, 1440), meta=np.nda...
          sos           (time, yh, xh)   float32   dask.array<chunksize=(1, 1142, 1440), meta=np.nda...
      ▸ Indexes:  (3)
      ▸ Attributes:  (20)
```

Why do I need to know the models name for a variable?



```
[79]: search = datastore.search(
          variable_standard_name=['sea_surface_salinity','sea_surface_temperature'],
      )

[80]: search.keys()

[80]: ['access_om3_mom6_2d_sos_1mon_mean_XXXX_XX.1mon',
       'access_om3_mom6_2d_tos_1mon_mean_XXXX_XX.1mon',
       'access_om3_mom6_2d_tos_1mon_min_XXXX_XX.1mon',
       'access_om3_mom6_scalar_1day_XXXX_XX.1day']

[81]: search = datastore.search(
          variable_standard_name=['sea_surface_salinity','sea_surface_temperature'],
          file_id='access_om3_mom6_scalar_1day_XXXX_XX'
      )
      search
```

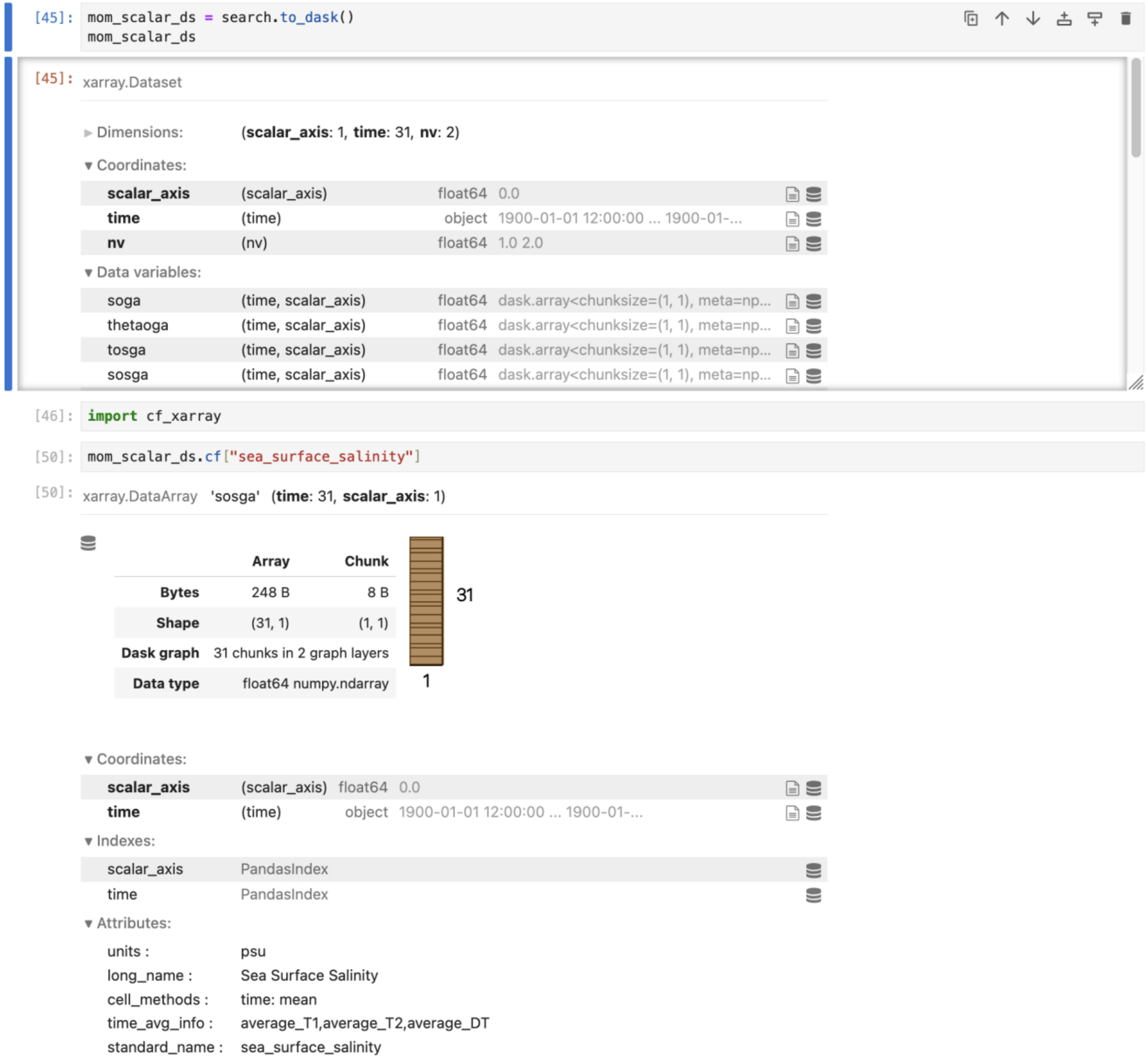
intake_esm_ds catalog with 1 dataset(s) from 1 asset(s):

	unique
path	1
realm	1
variable	25
frequency	1
start_date	1
end_date	1
variable_long_name	25
variable_standard_name	12
variable_cell_methods	1
variable_units	1
filename	1
file_id	1
derived_variable	0

See **Model Agnostic Analysis** at COSIMA-Recipes

https://cosima-recipes.readthedocs.io/en/latest/Tutorials/Model_Agnostic_Analysis.html

Why do I need to know the models name for a variable?



```
[45]: mom_scalar_ds = search.to_dask()
      mom_scalar_ds
```

```
[45]: xarray.Dataset

  ▸ Dimensions:      (scalar_axis: 1, time: 31, nv: 2)
  ▾ Coordinates:
      scalar_axis    (scalar_axis)      float64  0.0
      time           (time)             object   1900-01-01 12:00:00 ... 1900-01-...
      nv             (nv)               float64  1.0 2.0
  ▾ Data variables:
      soga           (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np...
      thetaoga       (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np...
      tosga          (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np...
      sosga          (time, scalar_axis)  float64  dask.array<chunksize=(1, 1), meta=np...
```

```
[46]: import cf_xarray
```

```
[50]: mom_scalar_ds.cf["sea_surface_salinity"]
```

```
[50]: xarray.DataArray  'sosga'  (time: 31, scalar_axis: 1)
```

	Array	Chunk
Bytes	248 B	8 B
Shape	(31, 1)	(1, 1)
Dask graph	31 chunks in 2 graph layers	
Data type	float64 numpy.ndarray	

```
  ▾ Coordinates:
      scalar_axis    (scalar_axis)  float64  0.0
      time           (time)         object   1900-01-01 12:00:00 ... 1900-01-...
  ▾ Indexes:
      scalar_axis    PandasIndex
      time           PandasIndex
  ▾ Attributes:
      units :          psu
      long_name :      Sea Surface Salinity
      cell_methods :   time: mean
      time_avg_info :  average_T1,average_T2,average_DT
      standard_name :  sea_surface_salinity
```

Ongoing Work

Panantarctic (GFDL-OM4) results are in xp65 environments, available for testing

Request additional datasets into the main catalog at

- https://github.com/ACCESS-NRI/access-nri-intake-catalog

List of cosima recipes converted to intake and open pull-requests:

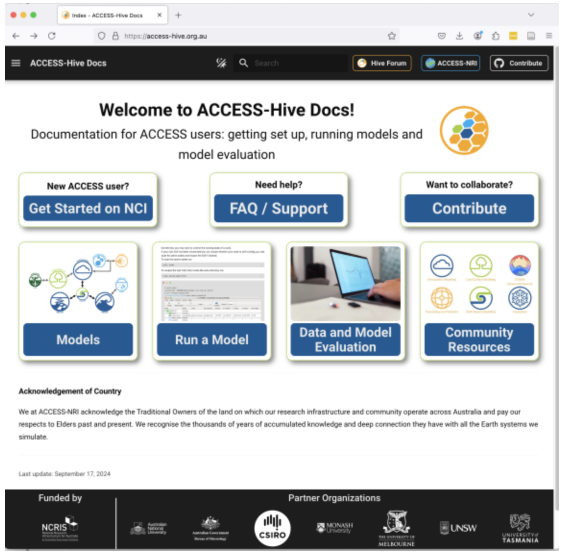
- https://github.com/COSIMA/cosima-recipes/issues/313

Updates & change since hh5 (xp65, conda/analysis3-25.02):

- Catalog now indexes grid variables
- `use_datastore/build-esm-datastore`
- More experiments and datasets indexed

NCI also have and add intake catalogs (e.g. ERA5, BARRA/BARPA, CMIP6)

For ACCESS documentation – see ACCESS-HIVE



Welcome to ACCESS-Hive Docs!

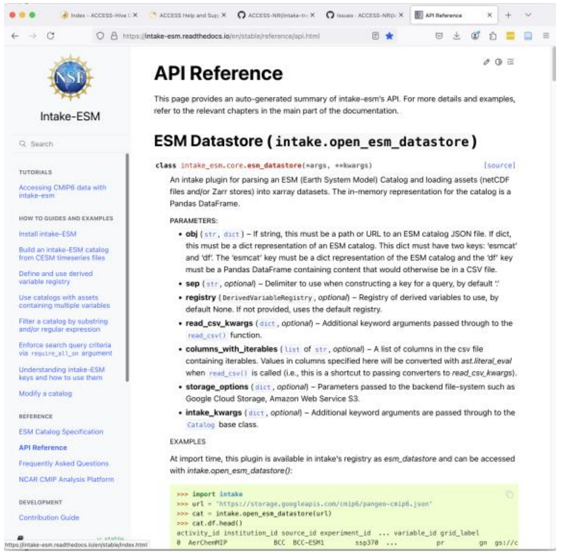
Documentation for ACCESS users: getting set up, running models and model evaluation

Acknowledgement of Country

We at ACCESS-NRI acknowledge the Traditional Owners of the land on which our research infrastructure and community operate across Australia and pay our respects to Elders past and present. We recognise the thousands of years of accumulated knowledge and deep connection they have with all the Earth systems we simulate.

Last update: September 17, 2024

For intake documentation – see intake-esm website


API Reference
https://intake-esm.readthedocs.io/en/stable/reference/api.html
Intake-ESM
Search
TUTORIALS
Accessing CMIP6 data with intake-esm
HOW TO GUIDES AND EXAMPLES
Install intake-ESM
Build an intake-ESM catalog from CESM timeseries files
Define and use derived variable registry
Use catalogs with assets containing multiple variables
Filter a catalog by substring and/or regular expression
Enforce search query criteria via require_all_on argument
Understanding intake-ESM keys and how to use them
Modify a catalog
REFERENCE
ESM Catalog Specification
API Reference
Frequently Asked Questions
NCAR CMIP Analysis Platform
DEVELOPMENT
Contribution Guide
https://intake-esm.readthedocs.io/en/stable/index.html

API Reference

This page provides an auto-generated summary of intake-esm's API. For more details and examples, refer to the relevant chapters in the main part of the documentation.

ESM Datastore (`intake.open_esm_datastore`)

`class intake_esm.core.esm_datastore(*args, **kwargs)` [source]

An intake plugin for parsing an ESM (Earth System Model) Catalog and loading assets (netCDF files and/or Zarr stores) into xarray datasets. The in-memory representation for the catalog is a Pandas DataFrame.

PARAMETERS:

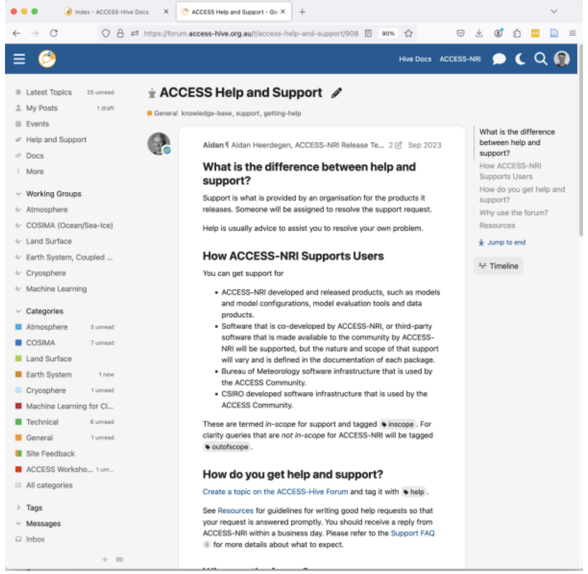
- **obj** (`str`, `dict`) – If string, this must be a path or URL to an ESM catalog JSON file. If dict, this must be a dict representation of an ESM catalog. This dict must have two keys: 'esmcat' and 'df'. The 'esmcat' key must be a dict representation of the ESM catalog and the 'df' key must be a Pandas DataFrame containing content that would otherwise be in a CSV file.
- **sep** (`str`, *optional*) – Delimiter to use when constructing a key for a query, by default '.'
- **registry** (`DerivedVariableRegistry`, *optional*) – Registry of derived variables to use, by default None. If not provided, uses the default registry.
- **read_csv_kwargs** (`dict`, *optional*) – Additional keyword arguments passed through to the `read_csv()` function.
- **columns_with_iterables** (`list` of `str`, *optional*) – A list of columns in the csv file containing iterables. Values in columns specified here will be converted with *ast.literal_eval* when `read_csv()` is called (i.e., this is a shortcut to passing converters to *read_csv_kwargs*).
- **storage_options** (`dict`, *optional*) – Parameters passed to the backend file-system such as Google Cloud Storage, Amazon Web Service S3.
- **intake_kwargs** (`dict`, *optional*) – Additional keyword arguments are passed through to the `Catalog` base class.

EXAMPLES

At import time, this plugin is available in intake's registry as *esm_datastore* and can be accessed with *intake.open_esm_datastore()*:

```
>>> import intake
>>> url = 'https://storage.googleapis.com/cmip6/pangeo-cmip6.json'
>>> cat = intake.open_esm_datastore(url)
>>> cat.df.head()
activity_id institution_id source_id experiment_id  ... variable_id grid_label
0  AerChemMIP           BCC  BCC-ESM1       ssp370  ...          pr         gn  gs://c
```

For general support – use ACCESS-HIVE FORUM



ACCESS Help and Support

General knowledge-base, support, getting-help

Aidan Aidan Heerdegen, ACCESS-NRI Release Te... 2 Sep 2023

What is the difference between help and support?

Support is what is provided by an organisation for the products it releases. Someone will be assigned to resolve the support request.

Help is usually advice to assist you to resolve your own problem.

How ACCESS-NRI Supports Users

You can get support for

- ACCESS-NRI developed and released products, such as models and model configurations, model evaluation tools and data products.
- Software that is co-developed by ACCESS-NRI, or third-party software that is made available to the community by ACCESS-NRI will be supported, but the nature and scope of that support will vary and is defined in the documentation of each package.
- Bureau of Meteorology software infrastructure that is used by the ACCESS Community.
- CSIRO developed software infrastructure that is used by the ACCESS Community.

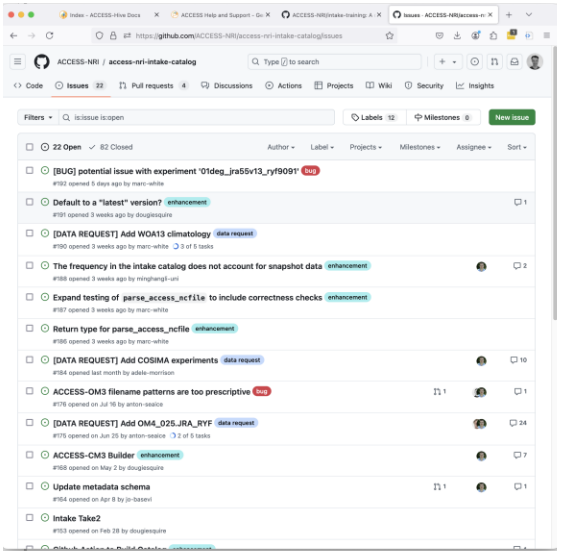
These are termed *in-scope* for support and tagged inscope. For clarity queries that are *not in-scope* for ACCESS-NRI will be tagged outofscope.

How do you get help and support?

Create a topic on the ACCESS-Hive Forum and tag it with help.

See Resources for guidelines for writing good help requests so that your request is answered promptly. You should receive a reply from ACCESS-NRI within a business day. Please refer to the Support FAQ 6 for more details about what to expect.

For bugs–use access-nri-intake-catalog github



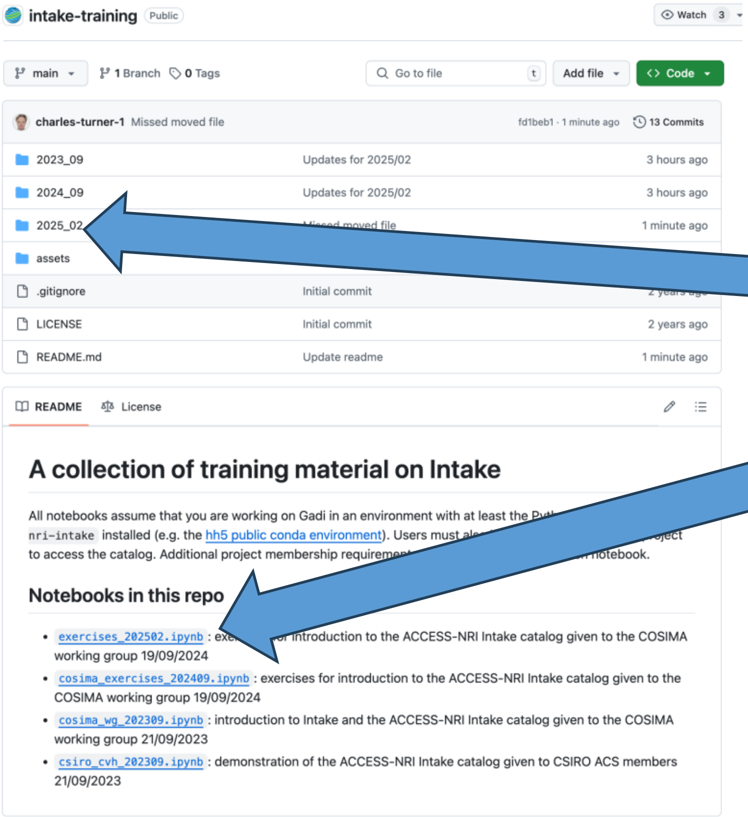
22 Open ✓ 82 Closed

- [BUG] potential issue with experiment '01deg_jra55v13_ryf9091' bug
  #192 opened 5 days ago by marc-white
- Default to a "latest" version? enhancement — 1
  #191 opened 3 weeks ago by dougiesquire
- [DATA REQUEST] Add WOA13 climatology data request
  #190 opened 3 weeks ago by marc-white 3 of 5 tasks
- The frequency in the intake catalog does not account for snapshot data enhancement — 2
  #188 opened 3 weeks ago by minghangli-uni
- Expand testing of `parse_access_ncfile` to include correctness checks enhancement
  #187 opened 3 weeks ago by marc-white
- Return type for parse_access_ncfile enhancement
  #186 opened 3 weeks ago by marc-white
- [DATA REQUEST] Add COSIMA experiments data request — 10
  #184 opened last month by adele-morrison
- ACCESS-OM3 filename patterns are too prescriptive bug — 1 — 1
  #176 opened on Jul 16 by anton-seaice
- [DATA REQUEST] Add OM4_025.JRA_RYF data request — 24
  #175 opened on Jun 25 by anton-seaice 2 of 5 tasks
- ACCESS-CM3 Builder enhancement — 7
  #168 opened on May 2 by dougiesquire
- Update metadata schema — 1 — 1
  #164 opened on Apr 8 by jo-basevi
- Intake Take2
  #153 opened on Feb 28 by dougiesquire

Some other resources

https://cosima-recipes.readthedocs.io/en/latest/Tutorials/ACCESS-NRI_Intake_Catalog.html

https://github.com/ACCESS-NRI/intake-training/



charles-turner-1 Missed moved file fd1beb1 · 1 minute ago 13 Commits

2023_09	Updates for 2025/02	3 hours ago
2024_09	Updates for 2025/02	3 hours ago
2025_02	Missed moved file	1 minute ago
assets		
.gitignore	Initial commit	2 years ago
LICENSE	Initial commit	2 years ago
README.md	Update readme	1 minute ago

A collection of training material on Intake

All notebooks assume that you are working on Gadi in an environment with at least the Pyt[illegible] `nri-intake` installed (e.g. the hh5 public conda environment). Users must al[illegible] oject to access the catalog. Additional project membership requiremen[illegible] notebook.

Notebooks in this repo

- `exercises_202502.ipynb` : exe[illegible] introduction to the ACCESS-NRI Intake catalog given to the COSIMA working group 19/09/2024
- `cosima_exercises_202409.ipynb` : exercises for introduction to the ACCESS-NRI Intake catalog given to the COSIMA working group 19/09/2024
- `cosima_wg_202309.ipynb` : introduction to Intake and the ACCESS-NRI Intake catalog given to the COSIMA working group 21/09/2023
- `csiro_cvh_202309.ipynb` : demonstration of the ACCESS-NRI Intake catalog given to CSIRO ACS members 21/09/2023

Todays exercises
can be found here

`use_datastore()` is not in hh5

Test set-up (very similar to a normal session for COSIMA Recipes):

Module directories

```
        /g/data/xp65/public/modules
```

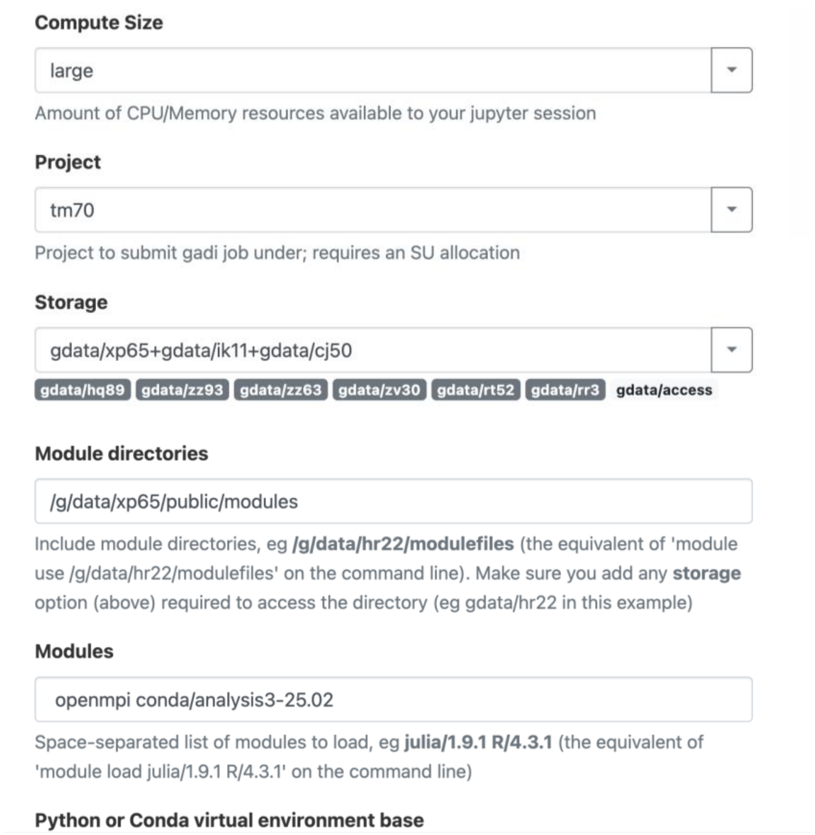
Modules

```
    openmpi conda/analysis3-25.02
```

Projects:

gdata/xp65+gdata/$PROJECT
(+gdata/hh5+gdata/ik11+gdata/cj50
+gdata/oi10+gdata/ol01+gdata/fs38+gdata/p73)

Compute Size of `large` or greater.



Compute Size

large

Amount of CPU/Memory resources available to your jupyter session

Project

tm70

Project to submit gadi job under; requires an SU allocation

Storage

gdata/xp65+gdata/ik11+gdata/cj50

gdata/hq89 gdata/zz93 gdata/zz63 gdata/zv30 gdata/rt52 gdata/rr3 gdata/access

Module directories

/g/data/xp65/public/modules

Include module directories, eg **/g/data/hr22/modulefiles** (the equivalent of 'module use /g/data/hr22/modulefiles' on the command line). Make sure you add any **storage** option (above) required to access the directory (eg gdata/hr22 in this example)

Modules

openmpi conda/analysis3-25.02

Space-separated list of modules to load, eg **julia/1.9.1 R/4.3.1** (the equivalent of 'module load julia/1.9.1 R/4.3.1' on the command line)

Python or Conda virtual environment base

Raise issues: https://github.com/ACCESS-NRI/ACCESS-Analysis-Conda

Exercises:

To follow the exercises:

```
$ git clone https://github.com/ACCESS-NRI/intake-training/
$ cd intake-training/2025_02
```

1. Make a datastore:

   datastore = use_datastore(…)

2. Find a single dataset:

```
    search = datastore.search(…)
```

Start a dask cluster

Open the dataset:

```
    search.to_dask()
```

3. Try another search which returns two variables.

```
    search = datastore.search(variable=[…,…])
    search.to_dataset_dict()
```

4. Try search with CF:

```
    datastore.search(variable_standard_name=‘’)
```

5. (Bonus) Run COSIMA recipe with your datastore, e.g:

https://github.com/COSIMA/cosima-recipes/blob/main/Recipes/Compare_SSH_model_obs.ipynb