intake-esm Documentation

Release 0.0.0

Intake-ESM developers

Aug 17, 2021

USING INTAKE-ESM

CHAPTER

ONE

BADGES

CI	
Docs	
Package	
License	
Citation	

CHAPTER

TWO

MOTIVATION

Computer simulations of the Earth's climate and weather generate huge amounts of data. These data are often persisted on HPC systems or in the cloud across multiple data assets of a variety of formats (netCDF, zarr, etc…). Finding, investigating, loading these data assets into compute-ready data containers costs time and effort. The data user needs to know what data sets are available, the attributes describing each data set, before loading a specific data set and analyzing it.

Finding, investigating, loading these assets into data array containers such as xarray can be a daunting task due to the large number of files a user may be interested in. Intake-esm aims to address these issues by providing necessary functionality for searching, discovering, data access/loading.

CHAPTER

THREE

OVERVIEW

`intake-esm` is a data cataloging utility built on top of intake, pandas, and xarray, and it's pretty awesome!

- Opening an ESM collection definition file: An ESM (Earth System Model) collection file is a JSON file that conforms to the ESM Collection Specification. When provided a link/path to an esm collection file, `intake-esm` establishes a link to a database (CSV file) that contains data assets locations and associated metadata (i.e., which experiment, model, the come from). The collection JSON file can be stored on a local filesystem or can be hosted on a remote server.

```
In [1]: import intake

In [2]: col_url = "https://storage.googleapis.com/cmip6/pangeo-cmip6.json"

In [3]: col = intake.open_esm_datastore(col_url)

In [4]: col
Out[4]: <pangeo-cmip6 catalog with 4287 dataset(s) from 282905 asset(s)>
```

- Search and Discovery: `intake-esm` provides functionality to execute queries against the catalog:

```
In [5]: col_subset = col.search(
   ...:     experiment_id=["historical", "ssp585"],
   ...:     table_id="Oyr",
   ...:     variable_id="o2",
   ...:     grid_label="gn",
   ...: )

In [6]: col_subset
Out[6]: <pangeo-cmip6 catalog with 18 dataset(s) from 138 asset(s)>
```

- Access: when the user is satisfied with the results of their query, they can ask `intake-esm` to load data assets (netCDF/HDF files and/or Zarr stores) into xarray datasets:

```
  In [7]: dset_dict = col_subset.to_dataset_dict(zarr_kwargs={"consolidated":␣
↪True})

  --> The keys in the returned dictionary of datasets are constructed as follows:
        'activity_id.institution_id.source_id.experiment_id.table_id.grid_label'
  || 100.00% [18/18 00:10<00:00]
```

See documentation for more information.

CHAPTER

FOUR

INSTALLATION

Intake-esm can be installed from PyPI with pip:

```
python -m pip install intake-esm
```

It is also available from `conda-forge` for conda installations:

```
conda install -c conda-forge intake-esm
```

CHAPTER

FIVE

GET IN TOUCH

- If you encounter any errors or problems with **pop-tools**, please open an issue at the GitHub main repository.
- If you have a question like "How do I find x?", ask on GitHub discussions. Please include a self-contained reproducible example if possible.

5.1 Installation

Intake-esm can be installed from PyPI with pip:

```
python -m pip install intake-esm
```

It is also available from `conda-forge` for conda installations:

```
conda install -c conda-forge intake-esm
```

5.2 User Guide

The intake-esm user guide introduces the main concepts required for accessing Earth Sytem Model (ESM) data catalogs and loading data assets into xarray containers. This guide gives an overview of the functionality available. The guide is split into core and tutorials & examples sections.

5.2.1 Overview

Intake-esm is a data cataloging utility built on top of intake, pandas, and xarray. Intake-esm aims to facilitate:

- the discovery of earth's climate and weather datasets.
- the ingestion of these datasets into xarray dataset containers.

It's basic usage is shown below. To begin, let's import `intake`:

```
import intake
```

Loading a catalog

At import time, intake-esm plugin is available in intake's registry as `esm_datastore` and can be accessed with `intake.open_esm_datastore()` function. For demonstration purposes, we are going to use the catalog for Community Earth System Model Large ensemble (CESM LENS) dataset publicly available in Amazon S3.

Note: You can learn more about CESM LENS dataset in AWS S3 here

You can load data from an ESM Catalog by providing the URL to valid ESM Catalog:

```
catalog_url = "https://ncar-cesm-lens.s3-us-west-2.amazonaws.com/catalogs/aws-cesm1-
↪le.json"
col = intake.open_esm_datastore(catalog_url)
col
```

```
<IPython.core.display.HTML object>
```

The summary above tells us that this catalog contains over 400 data assets. We can get more information on the individual data assets contained in the catalog by calling the underlying dataframe created when it is initialized:

```
col.df.head()
```

```
  variable                                long_name component experiment  \
0     FLNS             net longwave flux at surface       atm        20C
1    FLNSC    clearsky net longwave flux at surface       atm        20C
2     FLUT  upwelling longwave flux at top of model       atm        20C
3     FSNS                net solar flux at surface       atm        20C
4    FSNSC       clearsky net solar flux at surface       atm        20C

  frequency  vertical_levels spatial_domain units           start_time  \
0     daily              1.0         global  W/m2  1920-01-01 12:00:00
1     daily              1.0         global  W/m2  1920-01-01 12:00:00
2     daily              1.0         global  W/m2  1920-01-01 12:00:00
3     daily              1.0         global  W/m2  1920-01-01 12:00:00
4     daily              1.0         global  W/m2  1920-01-01 12:00:00

              end_time                                                path
0  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLNS....
1  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLNSC...
2  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLUT....
3  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FSNS....
4  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FSNSC...
```

Finding unique entries for individual columns

To get unique values for given columns in the catalog, intake-esm provides a *`unique()`* method. This method returns a dictionary containing count, and unique values:

```
col.unique(columns=["component", "frequency", "experiment"])
```

```
{'component': {'count': 5,
  'values': ['ice_nh', 'ocn', 'lnd', 'ice_sh', 'atm']},
 'frequency': {'count': 6,
```

(continues on next page)

(continued from previous page)

```
  'values': ['daily',
   'hourly6-1990-2005',
   'monthly',
   'hourly6-2071-2080',
   'static',
   'hourly6-2026-2035']},
 'experiment': {'count': 4, 'values': ['HIST', 'RCP85', '20C', 'CTRL']}}
```

Search

The *search()* method allows the user to perform a query on a catalog using keyword arguments. The keyword argument names must be the names of the columns in the catalog. The search method returns a subset of the catalog with all the entries that match the provided query.

Exact Match Keywords

By default, the *search()* method looks for exact matches

```
col_subset = col.search(
    component=["ice_nh", "lnd"],
    frequency=["monthly"],
    experiment=["20C", "HIST"],
)
col_subset.df
```

```
          variable                                          long_name  \
0             aice                                 ice area  (aggregate)
1               hi                           grid cell mean ice thickness
2             aice                                 ice area  (aggregate)
3               hi                           grid cell mean ice thickness
4             FSNO                    fraction of ground covered by snow
5           H2OSNO                              snow depth (liquid water)
6          QRUNOFF   total liquid runoff (does not include qsnwcpice)
7             RAIN                                       atmospheric rain
8             SNOW                                       atmospheric snow
9          SOILLIQ       soil liquid water (vegetated landunits only)
10  SOILWATER_10CM  soil liquid water + ice in top 10cm of soil (v...
11            FSNO                    fraction of ground covered by snow
12          H2OSNO                              snow depth (liquid water)
13         QRUNOFF   total liquid runoff (does not include qsnwcpice)
14            RAIN                                       atmospheric rain
15            SNOW                                       atmospheric snow
16         SOILLIQ       soil liquid water (vegetated landunits only)
17  SOILWATER_10CM  soil liquid water + ice in top 10cm of soil (v...

   component experiment frequency  vertical_levels spatial_domain     units  \
0     ice_nh        20C   monthly              1.0    artic_ocean         %
1     ice_nh        20C   monthly              1.0    artic_ocean         m
2     ice_nh       HIST   monthly              1.0    artic_ocean         %
3     ice_nh       HIST   monthly              1.0    artic_ocean         m
4        lnd        20C   monthly              1.0    global_land  unitless
5        lnd        20C   monthly              1.0    global_land        mm
6        lnd        20C   monthly              1.0    global_land      mm/s
```

(continues on next page)

(continued from previous page)

```
7         lnd         20C   monthly              1.0    global_land       mm/s
8         lnd         20C   monthly              1.0    global_land       mm/s
9         lnd         20C   monthly              1.0    global_land      kg/m2
10        lnd         20C   monthly              1.0    global_land      kg/m2
11        lnd        HIST   monthly              1.0    global_land   unitless
12        lnd        HIST   monthly              1.0    global_land         mm
13        lnd        HIST   monthly              1.0    global_land       mm/s
14        lnd        HIST   monthly              1.0    global_land       mm/s
15        lnd        HIST   monthly              1.0    global_land       mm/s
16        lnd        HIST   monthly              1.0    global_land      kg/m2
17        lnd        HIST   monthly              1.0    global_land      kg/m2

              start_time             end_time  \
0   1920-01-16 12:00:00  2005-12-16 12:00:00
1   1920-01-16 12:00:00  2005-12-16 12:00:00
2   1850-01-16 12:00:00  1919-12-16 12:00:00
3   1850-01-16 12:00:00  1919-12-16 12:00:00
4   1920-01-16 12:00:00  2005-12-16 12:00:00
5   1920-01-16 12:00:00  2005-12-16 12:00:00
6   1920-01-16 12:00:00  2005-12-16 12:00:00
7   1920-01-16 12:00:00  2005-12-16 12:00:00
8   1920-01-16 12:00:00  2005-12-16 12:00:00
9   1920-01-16 12:00:00  2005-12-16 12:00:00
10  1920-01-16 12:00:00  2005-12-16 12:00:00
11  1850-01-16 12:00:00  1919-12-16 12:00:00
12  1850-01-16 12:00:00  1919-12-16 12:00:00
13  1850-01-16 12:00:00  1919-12-16 12:00:00
14  1850-01-16 12:00:00  1919-12-16 12:00:00
15  1850-01-16 12:00:00  1919-12-16 12:00:00
16  1850-01-16 12:00:00  1919-12-16 12:00:00
17  1850-01-16 12:00:00  1919-12-16 12:00:00

                                                 path
0   s3://ncar-cesm-lens/ice_nh/monthly/cesmLE-20C-...
1   s3://ncar-cesm-lens/ice_nh/monthly/cesmLE-20C-...
2   s3://ncar-cesm-lens/ice_nh/monthly/cesmLE-HIST...
3   s3://ncar-cesm-lens/ice_nh/monthly/cesmLE-HIST...
4   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-FSN...
5   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-H2O...
6   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-QRU...
7   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-RAI...
8   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-SNO...
9   s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-SOI...
10  s3://ncar-cesm-lens/lnd/monthly/cesmLE-20C-SOI...
11  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-FS...
12  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-H2...
13  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-QR...
14  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-RA...
15  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-SN...
16  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-SO...
17  s3://ncar-cesm-lens/lnd/monthly/cesmLE-HIST-SO...
```

Substring matches

As pointed earlier, the search method looks for exact matches by default. However, with use of wildcards and/or regular expressions, we can find all items with a particular substring in a given column:

```
# Find all entries with `wind` in their variable long_name
col.search(long_name="wind*").df
```

```
     variable                                          long_name component  \
0        UBOT                      lowest model level zonal wind       atm
1   WSPDSRFAV  horizontal total wind speed average at the sur...       atm
2        UBOT                      lowest model level zonal wind       atm
3   WSPDSRFAV  horizontal total wind speed average at the sur...       atm
4        UBOT                      lowest model level zonal wind       atm
5        VBOT                 lowest model level meridional wind       atm
6           U                                          zonal wind       atm
7           U                                          zonal wind       atm
8           U                                          zonal wind       atm
9           U                                          zonal wind       atm
10          U                                          zonal wind       atm
11          U                                          zonal wind       atm
12          U                                          zonal wind       atm
13       TAUX                    windstress in grid-x direction       ocn
14      TAUX2                  windstress**2 in grid-x direction       ocn
15       TAUY                    windstress in grid-y direction       ocn
16      TAUY2                  windstress**2 in grid-y direction       ocn
17       TAUX                    windstress in grid-x direction       ocn
18      TAUX2                  windstress**2 in grid-x direction       ocn
19       TAUY                    windstress in grid-y direction       ocn
20      TAUY2                  windstress**2 in grid-y direction       ocn
21       TAUX                    windstress in grid-x direction       ocn
22      TAUX2                  windstress**2 in grid-x direction       ocn
23       TAUY                    windstress in grid-y direction       ocn
24      TAUY2                  windstress**2 in grid-y direction       ocn
25       TAUX                    windstress in grid-x direction       ocn
26      TAUX2                  windstress**2 in grid-x direction       ocn
27       TAUY                    windstress in grid-y direction       ocn
28      TAUY2                  windstress**2 in grid-y direction       ocn

   experiment          frequency  vertical_levels spatial_domain  \
0         20C              daily              1.0         global
1         20C              daily              1.0         global
2        HIST              daily              1.0         global
3        HIST              daily              1.0         global
4       RCP85              daily              1.0         global
5       RCP85              daily              1.0         global
6         20C  hourly6-1990-2005             30.0         global
7       RCP85  hourly6-2026-2035             30.0         global
8       RCP85  hourly6-2071-2080             30.0         global
9         20C            monthly             30.0         global
10       CTRL            monthly             30.0         global
11       HIST            monthly             30.0         global
12      RCP85            monthly             30.0         global
13        20C            monthly              1.0   global_ocean
14        20C            monthly              1.0   global_ocean
15        20C            monthly              1.0   global_ocean
16        20C            monthly              1.0   global_ocean
```

(continues on next page)

(continued from previous page)

```
17       CTRL            monthly              1.0   global_ocean
18       CTRL            monthly              1.0   global_ocean
19       CTRL            monthly              1.0   global_ocean
20       CTRL            monthly              1.0   global_ocean
21       HIST            monthly              1.0   global_ocean
22       HIST            monthly              1.0   global_ocean
23       HIST            monthly              1.0   global_ocean
24       HIST            monthly              1.0   global_ocean
25      RCP85            monthly              1.0   global_ocean
26      RCP85            monthly              1.0   global_ocean
27      RCP85            monthly              1.0   global_ocean
28      RCP85            monthly              1.0   global_ocean

                   units           start_time             end_time  \
0                    m/s  1920-01-01 12:00:00  2005-12-31 12:00:00
1                    m/s  1920-01-01 12:00:00  2005-12-31 12:00:00
2                    m/s  1850-01-01 12:00:00  1919-12-31 12:00:00
3                    m/s  1850-01-01 12:00:00  1919-12-31 12:00:00
4                    m/s  2006-01-01 12:00:00  2100-12-31 12:00:00
5                    m/s  2006-01-01 12:00:00  2100-12-31 12:00:00
6                    m/s  1990-01-01 00:00:00  2006-01-01 00:00:00
7                    m/s  2026-01-01 00:00:00  2036-01-01 00:00:00
8                    m/s  2071-01-01 00:00:00  2081-01-01 00:00:00
9                    m/s  1920-01-16 12:00:00  2005-12-16 12:00:00
10                   m/s  0400-01-16 12:00:00  2200-12-16 12:00:00
11                   m/s  1850-01-16 12:00:00  1919-12-16 12:00:00
12                   m/s  2006-01-16 12:00:00  2100-12-16 12:00:00
13    dyne/centimeter^2  1920-01-16 12:00:00  2005-12-16 12:00:00
14  dyne^2/centimeter^4  1920-01-16 12:00:00  2005-12-16 12:00:00
15    dyne/centimeter^2  1920-01-16 12:00:00  2005-12-16 12:00:00
16  dyne^2/centimeter^4  1920-01-16 12:00:00  2005-12-16 12:00:00
17    dyne/centimeter^2  0400-01-16 12:00:00  2200-12-16 12:00:00
18  dyne^2/centimeter^4  0400-01-16 12:00:00  2200-12-16 12:00:00
19    dyne/centimeter^2  0400-01-16 12:00:00  2200-12-16 12:00:00
20  dyne^2/centimeter^4  0400-01-16 12:00:00  2200-12-16 12:00:00
21    dyne/centimeter^2  1850-01-16 12:00:00  1919-12-16 12:00:00
22  dyne^2/centimeter^4  1850-01-16 12:00:00  1919-12-16 12:00:00
23    dyne/centimeter^2  1850-01-16 12:00:00  1919-12-16 12:00:00
24  dyne^2/centimeter^4  1850-01-16 12:00:00  1919-12-16 12:00:00
25    dyne/centimeter^2  2006-01-16 12:00:00  2100-12-16 12:00:00
26  dyne^2/centimeter^4  2006-01-16 12:00:00  2100-12-16 12:00:00
27    dyne/centimeter^2  2006-01-16 12:00:00  2100-12-16 12:00:00
28  dyne^2/centimeter^4  2006-01-16 12:00:00  2100-12-16 12:00:00

                                                    path
0   s3://ncar-cesm-lens/atm/daily/cesmLE-20C-UBOT....
1   s3://ncar-cesm-lens/atm/daily/cesmLE-20C-WSPDS...
2   s3://ncar-cesm-lens/atm/daily/cesmLE-HIST-UBOT...
3   s3://ncar-cesm-lens/atm/daily/cesmLE-HIST-WSPD...
4   s3://ncar-cesm-lens/atm/daily/cesmLE-RCP85-UBO...
5   s3://ncar-cesm-lens/atm/daily/cesmLE-RCP85-VBO...
6   s3://ncar-cesm-lens/atm/hourly6-1990-2005/cesm...
7   s3://ncar-cesm-lens/atm/hourly6-2026-2035/cesm...
8   s3://ncar-cesm-lens/atm/hourly6-2071-2080/cesm...
9   s3://ncar-cesm-lens/atm/monthly/cesmLE-20C-U.zarr
10  s3://ncar-cesm-lens/atm/monthly/cesmLE-CTRL-U....
11  s3://ncar-cesm-lens/atm/monthly/cesmLE-HIST-U....
```

(continues on next page)

(continued from previous page)

```
12  s3://ncar-cesm-lens/atm/monthly/cesmLE-RCP85-U...
13  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
14  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
15  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
16  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
17  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
18  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
19  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
20  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
21  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
22  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
23  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
24  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
25  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
26  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
27  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
28  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
```

```
# Find all entries whose variable long name starts with `wind`
col.search(long_name="^wind").df
```

```
   variable                          long_name component experiment frequency  \
0      TAUX     windstress in grid-x direction       ocn         20C   monthly
1     TAUX2  windstress**2 in grid-x direction       ocn         20C   monthly
2      TAUY     windstress in grid-y direction       ocn         20C   monthly
3     TAUY2  windstress**2 in grid-y direction       ocn         20C   monthly
4      TAUX     windstress in grid-x direction       ocn        CTRL   monthly
5     TAUX2  windstress**2 in grid-x direction       ocn        CTRL   monthly
6      TAUY     windstress in grid-y direction       ocn        CTRL   monthly
7     TAUY2  windstress**2 in grid-y direction       ocn        CTRL   monthly
8      TAUX     windstress in grid-x direction       ocn        HIST   monthly
9     TAUX2  windstress**2 in grid-x direction       ocn        HIST   monthly
10     TAUY     windstress in grid-y direction       ocn        HIST   monthly
11    TAUY2  windstress**2 in grid-y direction       ocn        HIST   monthly
12     TAUX     windstress in grid-x direction       ocn       RCP85   monthly
13    TAUX2  windstress**2 in grid-x direction       ocn       RCP85   monthly
14     TAUY     windstress in grid-y direction       ocn       RCP85   monthly
15    TAUY2  windstress**2 in grid-y direction       ocn       RCP85   monthly

    vertical_levels spatial_domain                units           start_time  \
0               1.0   global_ocean    dyne/centimeter^2  1920-01-16 12:00:00
1               1.0   global_ocean  dyne^2/centimeter^4  1920-01-16 12:00:00
2               1.0   global_ocean    dyne/centimeter^2  1920-01-16 12:00:00
3               1.0   global_ocean  dyne^2/centimeter^4  1920-01-16 12:00:00
4               1.0   global_ocean    dyne/centimeter^2  0400-01-16 12:00:00
5               1.0   global_ocean  dyne^2/centimeter^4  0400-01-16 12:00:00
6               1.0   global_ocean    dyne/centimeter^2  0400-01-16 12:00:00
7               1.0   global_ocean  dyne^2/centimeter^4  0400-01-16 12:00:00
8               1.0   global_ocean    dyne/centimeter^2  1850-01-16 12:00:00
9               1.0   global_ocean  dyne^2/centimeter^4  1850-01-16 12:00:00
10              1.0   global_ocean    dyne/centimeter^2  1850-01-16 12:00:00
11              1.0   global_ocean  dyne^2/centimeter^4  1850-01-16 12:00:00
12              1.0   global_ocean    dyne/centimeter^2  2006-01-16 12:00:00
13              1.0   global_ocean  dyne^2/centimeter^4  2006-01-16 12:00:00
14              1.0   global_ocean    dyne/centimeter^2  2006-01-16 12:00:00
15              1.0   global_ocean  dyne^2/centimeter^4  2006-01-16 12:00:00
```

(continues on next page)

(continued from previous page)

```
              end_time                                                path
0   2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
1   2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
2   2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
3   2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
4   2200-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
5   2200-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
6   2200-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
7   2200-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-CTRL-TA...
8   1919-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
9   1919-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
10  1919-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
11  1919-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-HIST-TA...
12  2100-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
13  2100-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
14  2100-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
15  2100-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-RCP85-T...
```

Loading datasets

Intake-esm implements convenience utilities for loading the query results into higher level xarray datasets. The logic for merging/concatenating the query results into higher level xarray datasets is provided in the input JSON file and is available under `.aggregation_info` property:

```
col.aggregation_info
```

```
AggregationInfo(groupby_attrs=['component', 'experiment', 'frequency'], variable_
↪column_name='variable', aggregations=[{'type': 'union', 'attribute_name': 'variable
↪', 'options': {'compat': 'override'}}], agg_columns=['variable'], aggregation_dict={
↪'variable': {'type': 'union', 'options': {'compat': 'override'}}})
```

```
col.aggregation_info.aggregations
```

```
[{'type': 'union',
  'attribute_name': 'variable',
  'options': {'compat': 'override'}}]
```

```
# Dataframe columns used to determine groups of compatible datasets.
col.aggregation_info.groupby_attrs  # or col.groupby_attrs
```

```
['component', 'experiment', 'frequency']
```

```
# List of columns used to merge/concatenate compatible multiple Dataset into a single␣
↪Dataset.
col.aggregation_info.agg_columns  # or col.agg_columns
```

```
['variable']
```

To load data assets into xarray datasets, we need to use the *to_dataset_dict()* method. This method returns a dictionary of aggregate xarray datasets as the name hints.

```
dset_dicts = col_subset.to_dataset_dict(zarr_kwargs={"consolidated": True})
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'component.experiment.frequency'
```

```
---------------------------------------------------------------------------
NoCredentialsError                        Traceback (most recent call last)
~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/merge_util.py in _open_
↪asset(path, data_format, zarr_kwargs, cdf_kwargs, preprocess, varname, requested_
↪variables)
    269         try:
--> 270             ds = xr.open_zarr(path, **zarr_kwargs)
    271         except Exception as exc:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/xarray/backends/zarr.py in open_zarr(store, group, synchronizer,␣
↪chunks, decode_cf, mask_and_scale, decode_times, concat_characters, decode_coords,␣
↪drop_variables, consolidated, overwrite_encoded_chunks, chunk_store, storage_
↪options, decode_timedelta, use_cftime, **kwargs)
    769
--> 770     ds = open_dataset(
    771         filename_or_obj=store,

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/xarray/backends/api.py in open_dataset(filename_or_obj, engine,␣
↪chunks, cache, decode_cf, mask_and_scale, decode_times, decode_timedelta, use_
↪cftime, concat_characters, decode_coords, drop_variables, backend_kwargs, *args,␣
↪**kwargs)
    496     overwrite_encoded_chunks = kwargs.pop("overwrite_encoded_chunks", None)
--> 497     backend_ds = backend.open_dataset(
    498         filename_or_obj,

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/xarray/backends/zarr.py in open_dataset(self, filename_or_obj, mask_
↪and_scale, decode_times, concat_characters, decode_coords, drop_variables, use_
↪cftime, decode_timedelta, group, mode, synchronizer, consolidated, chunk_store,␣
↪storage_options, stacklevel, lock)
    825         filename_or_obj = _normalize_path(filename_or_obj)
--> 826         store = ZarrStore.open_group(
    827             filename_or_obj,

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/xarray/backends/zarr.py in open_group(cls, store, mode, synchronizer,␣
↪group, consolidated, consolidate_on_close, chunk_store, storage_options, append_dim,
↪ write_region, safe_chunks, stacklevel)
    388             # TODO: an option to pass the metadata_key keyword
--> 389             zarr_group = zarr.open_consolidated(store, **open_kwargs)
    390         else:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/zarr/convenience.py in open_consolidated(store, metadata_key, mode,␣
↪**kwargs)
   1177     # setup metadata store
-> 1178     meta_store = ConsolidatedMetadataStore(store, metadata_key=metadata_key)
   1179
```

(continues on next page)

(continued from previous page)

```
~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/zarr/storage.py in __init__(self, store, metadata_key)
   2768         # retrieve consolidated metadata
-> 2769         meta = json_loads(store[metadata_key])
   2770

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/fsspec/mapping.py in __getitem__(self, key, default)
    132         try:
--> 133             result = self.fs.cat(k)
    134         except self.missing_exceptions:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/fsspec/asyn.py in wrapper(*args, **kwargs)
     87         self = obj or args[0]
---> 88         return sync(self.loop, func, *args, **kwargs)
     89

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/fsspec/asyn.py in sync(loop, func, timeout, *args, **kwargs)
     68     if isinstance(result[0], BaseException):
---> 69         raise result[0]
     70     return result[0]

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/fsspec/asyn.py in _runner(event, coro, result, timeout)
     24     try:
---> 25         result[0] = await coro
     26     except Exception as ex:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/fsspec/asyn.py in _cat(self, path, recursive, on_error, **kwargs)
    343             if ex:
--> 344                 raise ex
    345         if (

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/s3fs/core.py in _cat_file(self, path, version_id, start, end)
    850             head = {}
--> 851         resp = await self._call_s3(
    852             "get_object",

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/s3fs/core.py in _call_s3(self, method, *akwarglist, **kwargs)
    264                 err = e
--> 265         raise translate_boto_error(err)
    266

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/s3fs/core.py in _call_s3(self, method, *akwarglist, **kwargs)
    245             try:
--> 246                 out = await method(**additional_kwargs)
    247                 return out

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/client.py in _make_api_call(self, operation_name, api_
↪params)
```

(continues on next page)

(continued from previous page)

```
    140         else:
--> 141             http, parsed_response = await self._make_request(
    142                 operation_model, request_dict, request_context)

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/client.py in _make_request(self, operation_model, request_
↪dict, request_context)
    160         try:
--> 161             return await self._endpoint.make_request(operation_model, request_
↪dict)
    162         except Exception as e:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/endpoint.py in _send_request(self, request_dict,␣
↪operation_model)
     86         attempts = 1
---> 87         request = await self.create_request(request_dict, operation_model)
     88         context = request_dict['context']

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/endpoint.py in create_request(self, params, operation_
↪model)
     79                 op_name=operation_model.name)
---> 80             await self._event_emitter.emit(event_name, request=request,
     81                                            operation_name=operation_model.
↪name)

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/hooks.py in _emit(self, event_name, kwargs, stop_on_
↪response)
     26             if asyncio.iscoroutinefunction(handler):
---> 27                 response = await handler(**kwargs)
     28             else:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/signers.py in handler(self, operation_name, request,␣
↪**kwargs)
     15         # Don't call this method directly.
---> 16         return await self.sign(operation_name, request)
     17

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/aiobotocore/signers.py in sign(self, operation_name, request, region_
↪name, signing_type, expires_in, signing_name)
     62
---> 63             auth.add_auth(request)
     64

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/botocore/auth.py in add_auth(self, request)
    372         if self.credentials is None:
--> 373             raise NoCredentialsError()
    374         datetime_now = datetime.datetime.utcnow()

NoCredentialsError: Unable to locate credentials

The above exception was the direct cause of the following exception:
```

(continues on next page)

(continued from previous page)

```
OSError                                   Traceback (most recent call last)
/tmp/ipykernel_2336/728946501.py in <module>
----> 1 dset_dicts = col_subset.to_dataset_dict(zarr_kwargs={"consolidated": True})

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/core.py in to_dataset_dict(self,
↪ zarr_kwargs, cdf_kwargs, preprocess, storage_options, progressbar, aggregate)
    920             ]
    921             for i, task in enumerate(concurrent.futures.as_completed(future_
↪tasks)):
--> 922                 key, ds = task.result()
    923                 self._datasets[key] = ds
    924                 if self.progressbar:

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪concurrent/futures/_base.py in result(self, timeout)
    435                     raise CancelledError()
    436                 elif self._state == FINISHED:
--> 437                     return self.__get_result()
    438
    439                 self._condition.wait(timeout)

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪concurrent/futures/_base.py in __get_result(self)
    387         if self._exception:
    388             try:
--> 389                 raise self._exception
    390             finally:
    391                 # Break a reference cycle with the exception in self._
↪exception

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪concurrent/futures/thread.py in run(self)
     55
     56         try:
---> 57             result = self.fn(*self.args, **self.kwargs)
     58         except BaseException as exc:
     59             self.future.set_exception(exc)

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/core.py in _load_source(key,␣
↪source)
    906
    907         def _load_source(key, source):
--> 908             return key, source.to_dask()
    909
    910         sources = {key: source(**source_kwargs) for key, source in self.
↪items()}

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/source.py in to_dask(self)
    243     def to_dask(self):
    244         """Return xarray object (which will have chunks)"""
--> 245         self._load_metadata()
    246         return self._ds
    247
```

(continues on next page)

(continued from previous page)

```
~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake/source/base.py in _load_metadata(self)
    234         """load metadata only if needed"""
    235         if self._schema is None:
--> 236             self._schema = self._get_schema()
    237             self.dtype = self._schema.dtype
    238             self.shape = self._schema.shape

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/source.py in _get_schema(self)
    172
    173         if self._ds is None:
--> 174             self._open_dataset()
    175
    176             metadata = {

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/source.py in _open_dataset(self)
    224             for _, row in self.df.iterrows()
    225         ]
--> 226         datasets = dask.compute(*datasets)
    227         mapper_dict = dict(datasets)
    228         nd = create_nested_dict(self.df, self.path_column, self.aggregation_
↪columns)

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/base.py in compute(*args, **kwargs)
    566         postcomputes.append(x.__dask_postcompute__())
    567
--> 568     results = schedule(dsk, keys, **kwargs)
    569     return repack([f(r, *a) for r, (f, a) in zip(results, postcomputes)])
    570

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/threaded.py in get(dsk, result, cache, num_workers, pool,␣
↪**kwargs)
     77             pool = MultiprocessingPoolExecutor(pool)
     78
---> 79     results = get_async(
     80         pool.submit,
     81         pool._max_workers,

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/local.py in get_async(submit, num_workers, dsk, result, cache,␣
↪get_id, rerun_exceptions_locally, pack_exception, raise_exception, callbacks, dumps,
↪ loads, chunksize, **kwargs)
    512                             _execute_task(task, data)  # Re-execute locally
    513                         else:
--> 514                             raise_exception(exc, tb)
    515                     res, worker_id = loads(res_info)
    516                     state["cache"][key] = res

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/local.py in reraise(exc, tb)
    323     if exc.__traceback__ is not tb:
    324         raise exc.with_traceback(tb)
```

(continues on next page)

(continued from previous page)

```
--> 325     raise exc
    326
    327

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/local.py in execute_task(key, task_info, dumps, loads, get_id,␣
↪pack_exception)
    221     try:
    222         task, data = loads(task_info)
--> 223         result = _execute_task(task, data)
    224         id = get_id()
    225         result = dumps((result, id))

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/dask/core.py in _execute_task(arg, cache, dsk)
    119         # temporaries by their reference count and can execute certain
    120         # operations in-place.
--> 121         return func(*(_execute_task(a, cache) for a in args))
    122     elif not ishashable(arg):
    123         return arg

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/source.py in read_dataset(path,␣
↪data_format, storage_options, cdf_kwargs, zarr_kwargs, preprocess, varname)
    201             # replace path column with mapper (dependent on filesystem type)
    202             mapper = _path_to_mapper(path, storage_options, data_format)
--> 203             ds = _open_asset(
    204                 mapper,
    205                 data_format=data_format,

~/checkouts/readthedocs.org/user_builds/intake-esm/conda/v2021.8.17/lib/python3.8/
↪site-packages/intake_esm-0.0.0-py3.8.egg/intake_esm/merge_util.py in _open_
↪asset(path, data_format, zarr_kwargs, cdf_kwargs, preprocess, varname, requested_
↪variables)
    286             """
    287
--> 288             raise IOError(message) from exc
    289
    290     else:

OSError:
            Failed to open zarr store.

            *** Arguments passed to xarray.open_zarr() ***:

            - store: <fsspec.mapping.FSMap object at 0x7f179d86cb80>
            - kwargs: {'consolidated': True}

            *** fsspec options used ***:

            - root: ncar-cesm-lens/lnd/monthly/cesmLE-HIST-SOILWATER_10CM.zarr
            - protocol: ('s3', 's3a')

            ****************************************
```

```
[key for key in dset_dicts.keys()]
```

We can access a particular dataset as follows:

```
ds = dset_dicts["lnd.20C.monthly"]
print(ds)
```

Let's create a quick plot for a slice of the data:

```
ds.SNOW.isel(time=0, member_id=range(1, 24, 4)).plot(col="member_id", col_wrap=3,␣
↪robust=True)
```

```
import intake_esm  # just to display version information

intake_esm.show_versions()
```

5.2.2 Search and Discovery

Intake-esm provides functionality to execute queries against the catalog. This notebook provided a more in-depth treatment of the search API in intake-esm, with detailed information that you can refer to when needed.

```
import warnings

warnings.filterwarnings("ignore")
import intake
```

```
catalog_url = "https://ncar-cesm-lens.s3-us-west-2.amazonaws.com/catalogs/aws-cesm1-
↪le.json"
col = intake.open_esm_datastore(catalog_url)
col
```

```
<IPython.core.display.HTML object>
```

```
col.df.head()
```

```
  variable                                long_name component experiment  \
0     FLNS             net longwave flux at surface       atm        20C
1    FLNSC    clearsky net longwave flux at surface       atm        20C
2     FLUT  upwelling longwave flux at top of model       atm        20C
3     FSNS               net solar flux at surface       atm        20C
4    FSNSC       clearsky net solar flux at surface       atm        20C

  frequency  vertical_levels spatial_domain units           start_time  \
0     daily              1.0         global  W/m2  1920-01-01 12:00:00
1     daily              1.0         global  W/m2  1920-01-01 12:00:00
2     daily              1.0         global  W/m2  1920-01-01 12:00:00
3     daily              1.0         global  W/m2  1920-01-01 12:00:00
4     daily              1.0         global  W/m2  1920-01-01 12:00:00

              end_time                                                path
0  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLNS....
1  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLNSC...
2  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FLUT....
```

(continues on next page)

(continued from previous page)

```
3  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FSNS....
4  2005-12-31 12:00:00  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-FSNSC...
```

Exact Match Keywords

The *search()* method allows the user to perform a query on a catalog using keyword arguments. The keyword argument names must be the names of the columns in the catalog. By default, the *search()* method looks for exact matches, and is case sensitive:

```
col.search(experiment="20C", long_name="wind").df
```

```
Empty DataFrame
Columns: [variable, long_name, component, experiment, frequency, vertical_levels,␣
↪spatial_domain, units, start_time, end_time, path]
Index: []
```

As you can see, the example above returns an empty catalog.

Substring Matches

In some cases, you may not know the exact term to look for. For such cases, inkake-esm supports searching for substring matches. With use of wildcards and/or regular expressions, we can find all items with a particular substring in a given column. Let's search for:

- entries from `experiment` = '20C'
- all entries whose variable long name **contains** `wind`

```
col.search(experiment="20C", long_name="wind*").df
```

```
    variable                                          long_name component  \
0       UBOT                     lowest model level zonal wind       atm
1  WSPDSRFAV  horizontal total wind speed average at the sur...       atm
2          U                                        zonal wind       atm
3          U                                        zonal wind       atm
4       TAUX                    windstress in grid-x direction       ocn
5      TAUX2                  windstress**2 in grid-x direction       ocn
6       TAUY                    windstress in grid-y direction       ocn
7      TAUY2                  windstress**2 in grid-y direction       ocn

  experiment          frequency  vertical_levels spatial_domain  \
0        20C              daily              1.0         global
1        20C              daily              1.0         global
2        20C  hourly6-1990-2005             30.0         global
3        20C            monthly             30.0         global
4        20C            monthly              1.0   global_ocean
5        20C            monthly              1.0   global_ocean
6        20C            monthly              1.0   global_ocean
7        20C            monthly              1.0   global_ocean

                 units           start_time             end_time  \
0                  m/s  1920-01-01 12:00:00  2005-12-31 12:00:00
1                  m/s  1920-01-01 12:00:00  2005-12-31 12:00:00
```

(continues on next page)

(continued from previous page)

```
2                  m/s  1990-01-01 00:00:00  2006-01-01 00:00:00
3                  m/s  1920-01-16 12:00:00  2005-12-16 12:00:00
4    dyne/centimeter^2  1920-01-16 12:00:00  2005-12-16 12:00:00
5  dyne^2/centimeter^4  1920-01-16 12:00:00  2005-12-16 12:00:00
6    dyne/centimeter^2  1920-01-16 12:00:00  2005-12-16 12:00:00
7  dyne^2/centimeter^4  1920-01-16 12:00:00  2005-12-16 12:00:00

                                          path
0  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-UBOT....
1  s3://ncar-cesm-lens/atm/daily/cesmLE-20C-WSPDS...
2  s3://ncar-cesm-lens/atm/hourly6-1990-2005/cesm...
3  s3://ncar-cesm-lens/atm/monthly/cesmLE-20C-U.zarr
4  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
5  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
6  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
7  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
```

Now, let's search for:

- entries from `experiment` = '20C'
- all entries whose variable long name **starts** with `wind`

```
col.search(experiment="20C", long_name="^wind").df
```

```
  variable                          long_name component experiment frequency  \
0     TAUX     windstress in grid-x direction       ocn        20C   monthly
1    TAUX2  windstress**2 in grid-x direction       ocn        20C   monthly
2     TAUY     windstress in grid-y direction       ocn        20C   monthly
3    TAUY2  windstress**2 in grid-y direction       ocn        20C   monthly

   vertical_levels spatial_domain                units           start_time  \
0              1.0   global_ocean    dyne/centimeter^2  1920-01-16 12:00:00
1              1.0   global_ocean  dyne^2/centimeter^4  1920-01-16 12:00:00
2              1.0   global_ocean    dyne/centimeter^2  1920-01-16 12:00:00
3              1.0   global_ocean  dyne^2/centimeter^4  1920-01-16 12:00:00

             end_time                                                path
0  2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
1  2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
2  2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
3  2005-12-16 12:00:00  s3://ncar-cesm-lens/ocn/monthly/cesmLE-20C-TAU...
```

Enforce Query Criteria via `require_all_on argument`

By default intake-esm's *`search()`* method returns entries that fulfill **any** of the criteria specified in the query. Intake-esm can return entries that fulfill **all** query criteria when the user supplies the `require_all_on` argument. The `require_all_on` parameter can be a dataframe column or a list of dataframe columns across which all elements must satisfy the query criteria. The `require_all_on` argument is best explained with the following example.

Let's define a query for our collection that requests multiple variable_ids and multiple experiment_ids from the Omon table_id, all from 3 different source_ids:

```
catalog_url = "https://storage.googleapis.com/cmip6/pangeo-cmip6.json"
col = intake.open_esm_datastore(catalog_url)
col
```

```
<IPython.core.display.HTML object>
```

```
# Define our query
query = dict(
    variable_id=["thetao", "o2"],
    experiment_id=["historical", "ssp245", "ssp585"],
    table_id=["Omon"],
    source_id=["ACCESS-ESM1-5", "AWI-CM-1-1-MR", "FGOALS-f3-L"],
)
```

Now, let's use this query to search for all assets in the collection that satisfy any combination of these requests (i.e., with `require_all_on=None`, which is the default):

```
col_subset = col.search(**query)
col_subset
```

```
<IPython.core.display.HTML object>
```

```
# Group by `source_id` and count unique values for a few columns
col_subset.df.groupby("source_id")[["experiment_id", "variable_id", "table_id"]].
↪nunique()
```

```
              experiment_id  variable_id  table_id
source_id
ACCESS-ESM1-5             3            2         1
AWI-CM-1-1-MR             3            1         1
FGOALS-f3-L               3            1         1
```

As you can see, the search results above include source_ids for which we only have one of the two variables, and one or two of the three experiments.

We can tell intake-esm to discard any source_id that doesn't have both variables `["thetao", "o2"]` and all three experiments `["historical", "ssp245", "ssp585"]` by passing `require_all_on=["source_id"]` to the search method:

```
col_subset = col.search(require_all_on=["source_id"], **query)
col_subset
```

```
<IPython.core.display.HTML object>
```

```
col_subset.df.groupby("source_id")[["experiment_id", "variable_id", "table_id"]].
↪nunique()
```

```
              experiment_id  variable_id  table_id
source_id
ACCESS-ESM1-5             3            2         1
```

Notice that with the `require_all_on=["source_id"]` option, the only source_id that was returned by our query was the source_id for which all of the variables and experiments were found.

```
import intake_esm  # just to display version information

intake_esm.show_versions()
```

```
INSTALLED VERSIONS
------------------

cftime: 1.5.0
dask: 2021.08.0
fastprogress: 0.2.7
fsspec: 2021.07.0
gcsfs: 2021.07.0
intake: 0.6.3
intake_esm: 0.0.0
netCDF4: 1.5.7
pandas: 1.3.2
requests: 2.26.0
s3fs: 2021.07.0
xarray: 0.19.0
zarr: 2.8.3
```

5.2.3 Working with multi-variable assets

In addition to catalogs of data assets (files) in time-series (single-variable) format, intake-esm supports catalogs with data assets in time-slice (history) format and/or files with multiple variables. For intake-esm to properly work with multi-variable assets,

- the `variable_column` of the catalog must contain iterables (list, tuple, set) of values.
- the user must specifiy a dictionary of functions for converting values in certain columns into iterables. This is done via the `csv_kwargs` argument.

In the example below, we are are going to use the following catalog to demonstrate how to work with multi-variable assets:

```
# Look at the catalog on disk
!cat multi-variable-catalog.csv
```

```
experiment,case,component,stream,variable,member_id,path,time_range
CTRL,b.e11.B1850C5CN.f09_g16.005,ocn,pop.h,"['SHF', 'REGION_MASK', 'ANGLE', 'DXU',
↪'KMT', 'NO2', 'O2']",5,../../../tests/sample_data/cesm-multi-variables/b.e11.
↪B1850C5CN.f09_g16.005.pop.h.SHF-NO2-O2.050001-050012.nc,050001-050012
CTRL,b.e11.B1850C5CN.f09_g16.005,ocn,pop.h,"['SHF', 'REGION_MASK', 'ANGLE', 'DXU',
↪'KMT', 'NO2', 'O2']",5,../../../tests/sample_data/cesm-multi-variables/b.e11.
↪B1850C5CN.f09_g16.005.pop.h.SHF-NO2-O2.050101-050112.nc,050101-050112
CTRL,b.e11.B1850C5CN.f09_g16.005,ocn,pop.h,"['SHF', 'REGION_MASK', 'ANGLE', 'DXU',
↪'KMT', 'NO2', 'PO4']",5,../../../tests/sample_data/cesm-multi-variables/b.e11.
↪B1850C5CN.f09_g16.005.pop.h.SHF-NO2-PO4.050001-050012.nc,050001-050012
CTRL,b.e11.B1850C5CN.f09_g16.005,ocn,pop.h,"['SHF', 'REGION_MASK', 'ANGLE', 'DXU',
↪'KMT', 'NO2', 'PO4']",5,../../../tests/sample_data/cesm-multi-variables/b.e11.
↪B1850C5CN.f09_g16.005.pop.h.SHF-NO2-PO4.050101-050112.nc,050101-050112
CTRL,b.e11.B1850C5CN.f09_g16.005,ocn,pop.h,"['SHF', 'REGION_MASK', 'ANGLE', 'DXU',
↪'KMT', 'TEMP', 'SiO3']",5,../../../tests/sample_data/cesm-multi-variables/b.e11.
↪B1850C5CN.f09_g16.005.pop.h.SHF-TEMP-SiO3.050001-050012.nc,050001-050012
```

As you can see, the variable column contains a list of varibles, and this list was serialized as a string: `"['SHF', 'REGION_MASK', 'ANGLE', 'DXU', 'KMT', 'NO2', 'O2']"`.

Loading a catalog

To load a catalog with multiple variable files, we must pass additional information to `open_esm_datastore` via the `csv_kwargs` argument. We are going to specify a dictionary of functions for converting values in `variable` column into iterables. We use the `literal_eval` function from the standard `ast` module:

```
import ast

import intake
```

```
col = intake.open_esm_datastore(
    "multi-variable-collection.json",
    csv_kwargs={"converters": {"variable": ast.literal_eval}},
)
col
```

```
<IPython.core.display.HTML object>
```

```
col.df.head()
```

```
  experiment                          case component stream  \
0       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
1       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
2       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
3       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
4       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h

                                        variable  member_id  \
0     (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, O2)          5
1     (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, O2)          5
2    (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, PO4)          5
3    (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, PO4)          5
4  (SHF, REGION_MASK, ANGLE, DXU, KMT, TEMP, SiO3)          5

                                             path     time_range
0  ../../../tests/sample_data/cesm-multi-variable...  050001-050012
1  ../../../tests/sample_data/cesm-multi-variable...  050101-050112
2  ../../../tests/sample_data/cesm-multi-variable...  050001-050012
3  ../../../tests/sample_data/cesm-multi-variable...  050101-050112
4  ../../../tests/sample_data/cesm-multi-variable...  050001-050012
```

The in-memory representation of the catalog contains `variable` with tuple of values. To confirm that intake-esm has registered this catalog with multiple variable assets, we can the `._multiple_variable_assets` property:

```
col._multiple_variable_assets
```

```
True
```

Searching

The search functionatilty works in the same way:

```
col_subset = col.search(variable=["O2", "SiO3"])
col_subset.df
```

```
  experiment                          case component stream  \
0       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
1       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h
2       CTRL  b.e11.B1850C5CN.f09_g16.005       ocn  pop.h

                                       variable  member_id  \
0     (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, O2)          5
1     (SHF, REGION_MASK, ANGLE, DXU, KMT, NO2, O2)          5
2  (SHF, REGION_MASK, ANGLE, DXU, KMT, TEMP, SiO3)          5

                                                path     time_range
0  ../../../tests/sample_data/cesm-multi-variable...  050001-050012
1  ../../../tests/sample_data/cesm-multi-variable...  050101-050112
2  ../../../tests/sample_data/cesm-multi-variable...  050001-050012
```

Loading assets into xarray datasets

Loading data assets into xarray datasets works in the same way too:

```
col_subset.to_dataset_dict(cdf_kwargs={})
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'component.experiment.stream'
```

```
<IPython.core.display.HTML object>
```

```
{'ocn.CTRL.pop.h': <xarray.Dataset>
 Dimensions:    (time: 24, member_id: 1, nlat: 2, nlon: 2)
 Coordinates:
   * time       (time) object 0500-02-01 00:00:00 ... 0502-02-01 00:00:00
     TLAT       (nlat, nlon) float64 dask.array<chunksize=(2, 2), meta=np.ndarray>
     TLONG      (nlat, nlon) float64 dask.array<chunksize=(2, 2), meta=np.ndarray>
     ULAT       (nlat, nlon) float64 dask.array<chunksize=(2, 2), meta=np.ndarray>
     ULONG      (nlat, nlon) float64 dask.array<chunksize=(2, 2), meta=np.ndarray>
   * member_id  (member_id) int64 5
 Dimensions without coordinates: nlat, nlon
 Data variables:
     O2         (member_id, time, nlat, nlon) float32 dask.array<chunksize=(1, 12, 2,␣
↪2), meta=np.ndarray>
     SiO3       (member_id, time, nlat, nlon) float32 dask.array<chunksize=(1, 24, 2,␣
↪2), meta=np.ndarray>
 Attributes: (12/16)
     start_time:               This dataset was created on 2013-05-28 at 02:4...
     revision:                 $Id: tavg.F90 41939 2012-11-14 16:37:23Z mlevy...
     tavg_sum:                 2678400.0
     tavg_sum_qflux:           2678400.0
     NCO:                      4.3.4
     title:                    b.e11.B1850C5CN.f09_g16.005
```

(continues on next page)

(continued from previous page)

```
    ...                          ...
    cell_methods:                cell_methods = time: mean ==> the variable val...
    nco_openmp_thread_number:    1
    Conventions:                 CF-1.0; http://www.cgd.ucar.edu/cms/eaton/netc...
    intake_esm_varname:          O2\nSiO3
    calendar:                    All years have exactly  365 days.
    intake_esm_dataset_key:      ocn.CTRL.pop.h}
```

```
import intake_esm  # just to display version information

intake_esm.show_versions()
```

```
INSTALLED VERSIONS
------------------

cftime: 1.5.0
dask: 2021.08.0
fastprogress: 0.2.7
fsspec: 2021.07.0
gcsfs: 2021.07.0
intake: 0.6.3
intake_esm: 0.0.0
netCDF4: 1.5.7
pandas: 1.3.2
requests: 2.26.0
s3fs: 2021.07.0
xarray: 0.19.0
zarr: 2.8.3
```

5.2.4 Load CMIP6 Data with Intake ESM

This notebook demonstrates how to access Google Cloud CMIP6 data using intake-esm.

Loading a catalog

```
import warnings

warnings.filterwarnings("ignore")
import intake
```

```
url = "https://storage.googleapis.com/cmip6/pangeo-cmip6.json"
col = intake.open_esm_datastore(url)
col
```

```
Matplotlib is building the font cache; this may take a moment.
```

```
<IPython.core.display.HTML object>
```

The summary above tells us that this catalog contains over 268,000 data assets. We can get more information on the individual data assets contained in the catalog by calling the underlying dataframe created when it is initialized:

Catalog Contents

```
col.df.head()
```

```
  activity_id institution_id     source_id       experiment_id member_id  \
0  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
1  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
2  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
3  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
4  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1

  table_id variable_id grid_label  \
0     Amon         hus         gn
1     Amon        rsdt         gn
2     Amon         prw         gn
3     Amon        rlus         gn
4     Amon        rlds         gn

                                          zstore  dcpp_init_year   version
0  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
1  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
2  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
3  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
4  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
```

The first data asset listed in the catalog contains:

- the ambient aerosol optical thickness at 550nm (`variable_id='od550aer'`), as a function of latitude, longitude, time,
- in an individual climate model experiment with the Taiwan Earth System Model 1.0 model (`source_id='TaiESM1'`),
- forced by the *Historical transient with SSTs prescribed from historical* experiment (`experiment_id='histSST'`),
- developed by the Taiwan Research Center for Environmental Changes (`instution_id='AS-RCEC'`),
- run as part of the Aerosols and Chemistry Model Intercomparison Project (`activity_id='AerChemMIP'`)

And is located in Google Cloud Storage at `gs://cmip6/AerChemMIP/AS-RCEC/TaiESM1/histSST/r1i1p1f1/AERmon/od550aer/gn/`.

Finding unique entries

Let's query the data to see what models (`source_id`), experiments (`experiment_id`) and temporal frequencies (`table_id`) are available.

```
import pprint

uni_dict = col.unique(["source_id", "experiment_id", "table_id"])
pprint.pprint(uni_dict, compact=True)
```

```
{'experiment_id': {'count': 172,
                   'values': ['histSST-piNTCF', 'abrupt-solm4p',
                              'piClim-2xdust', 'aqua-p4K-lwoff', 'esm-hist',
                              'r7i1p1f1', 'dcppC-amv-Trop-neg',
```

(continues on next page)

(continued from previous page)

```
                                'faf-heat-NA50pct', 'ssp245-covid',
                                'ssp245-cov-strgreen', 'histSST-1950HC',
                                'dcppC-pac-pacemaker', 'ssp370SST-lowCH4',
                                'piClim-histghg', 'ssp119', 'dcppA-hindcast',
                                'abrupt-solp4p', 'ssp460', 'ssp370SST-ssp126Lu',
                                'piClim-NTCF', 'hist-resIPO', 'aqua-4xCO2',
                                'piClim-anthro', 'ssp585-bgc', 'r4i1p1f1',
                                'esm-piControl', 'dcppC-amv-ExTrop-neg',
                                'esm-pi-cdr-pulse', 'control-1950',
                                'pdSST-pdSICSIT', 'faf-stress', 'rcp45-cmip5',
                                'hist-1950HC', 'lgm', 'ssp370-lowNTCF',
                                'piClim-O3', 'faf-passiveheat', 'ssp245-GHG',
                                'histSST-piAer', 'abrupt-0p5xCO2', 'faf-heat',
                                'hist-totalO3', 'piClim-lu', 'piClim-2xfire',
                                'aqua-p4K', 'piClim-BC', 'piClim-NOx',
                                'piClim-ghg', 'dcppC-pac-control', 'hist-piAer',
                                'pa-piAntSIC', 'abrupt-4xCO2', 'piClim-aer',
                                'dcppC-ipv-NexTrop-neg', 'land-hist-altStartYear',
                                'pdSST-piArcSIC', 'lig127k', 'midHolocene',
                                'highresSST-present', 'piClim-histaer',
                                'dcppC-amv-ExTrop-pos', 'amip-4xCO2',
                                'aqua-control', 'piClim-histnat',
                                'ssp370-ssp126Lu', 'hist-bgc',
                                'dcppC-amv-Trop-pos', 'pdSST-pdSIC',
                                '1pctCO2-rad', 'dcppC-hindcast-noElChichon',
                                'amip-p4K', 'esm-pi-CO2pulse', 'dcppC-ipv-pos',
                                'piControl-spinup', 'ssp245-cov-modgreen',
                                'esm-ssp585', 'histSST-piCH4', 'hist-CO2',
                                'land-hist', 'piControl', 'histSST-piO3',
                                'pdSST-piAntSIC', 'pdSST-futArcSICSIT',
                                'ssp245-cov-fossil', 'piClim-4xCO2',
                                'abrupt-2xCO2', '1pctCO2-bgc', 'piClim-control',
                                'aqua-control-lwoff', 'futSST-pdSIC',
                                'piClim-SO2', 'hist-1950', 'hist-volc',
                                'past1000', 'ssp370', 'amip-hist',
                                'pdSST-futArcSIC', 'historical-cmip5',
                                'dcppC-hindcast-noPinatubo', 'piClim-OC',
                                'amip-future4K', 'hist-aer', 'pa-pdSIC',
                                'ssp370SST', 'dcppC-ipv-NexTrop-pos',
                                'esm-ssp585-ssp126Lu', 'pa-futAntSIC',
                                'piClim-HC', 'dcppC-amv-neg', 'ssp585',
                                'ssp534-over', 'dcppA-assim', 'faf-heat-NA0pct',
                                'piClim-VOC', 'land-noLu', 'deforest-globe',
                                'piClim-N2O', 'dcppC-amv-pos', 'pdSST-futAntSIC',
                                'ssp126-ssp370Lu', 'piClim-CH4', 'dcppC-ipv-neg',
                                'hist-piNTCF', 'r5i1p1f1', 'pdSST-futOkhotskSIC',
                                'histSST', 'pdSST-futBKSeasSIC',
                                'esm-piControl-spinup', 'piClim-2xDMS',
                                'ssp370SST-lowNTCF', 'ssp126', 'ssp370pdSST',
                                'historical', 'dcppC-hindcast-noAgung',
                                'pa-futArcSIC', 'r6i1p1f1', 'piSST-piSIC',
                                'rcp85-cmip5', 'ssp434', 'piClim-histall',
                                'faf-water', 'ssp245', 'dcppC-atl-control',
                                'hist-sol', 'historical-ext', 'piClim-2xNOx',
                                'hist-nat-cmip5', 'piSST-pdSIC', 'piClim-2xss',
                                'hist-nat', 'piControl-cmip5', 'pa-piArcSIC',
                                'highresSST-future', 'hist-noLu', 'amip-lwoff',
```

(continues on next page)

(continued from previous page)

```
                              'hist-stratO3', '1pctCO2', 'hist-aer-cmip5',
                              'amip', 'hist-GHG', 'ssp245-aer', 'amip-m4K',
                              'dcppC-atl-pacemaker', 'amip-p4K-lwoff',
                              'rcp26-cmip5', '1pctCO2-cdr', 'omip1', 'faf-all',
                              'ssp245-nat', 'hist-GHG-cmip5', 'ssp245-stratO3',
                              'piClim-2xVOC']},
 'source_id': {'count': 87,
               'values': ['ECMWF-IFS-LR', 'EC-Earth3P-VHR', 'UKESM1-0-LL',
                          'CAMS-CSM1-0', 'EC-Earth3', 'CNRM-CM6-1-HR',
                          'CMCC-CM2-HR4', 'EC-Earth3-AerChem', 'CESM2-FV2',
                          'CNRM-ESM2-1', 'GFDL-CM4C192', 'INM-CM4-8',
                          'AWI-ESM-1-1-LR', 'CAS-ESM2-0', 'GFDL-ESM4',
                          'CMCC-CM2-SR5', 'MIROC-ES2H', 'FGOALS-g3',
                          'GISS-E2-1-G-CC', 'MRI-AGCM3-2-H', 'TaiESM1',
                          'GISS-E2-1-H', 'CMCC-CM2-VHR4', 'CESM2-WACCM',
                          'MPI-ESM1-2-LR', 'HadGEM3-GC31-LL', 'CanESM5',
                          'HadGEM3-GC31-HM', 'AWI-CM-1-1-MR', 'CanESM5-CanOE',
                          'MPI-ESM1-2-XR', 'BCC-CSM2-MR', 'EC-Earth3-Veg',
                          'FIO-ESM-2-0', 'E3SM-1-1-ECA', 'MPI-ESM1-2-HR',
                          'CESM2', 'ACCESS-CM2', 'EC-Earth3-CC', 'NESM3',
                          'CESM1-1-CAM5-CMIP5', 'MRI-AGCM3-2-S', 'ECMWF-IFS-HR',
                          'BCC-ESM1', 'NorCPM1', 'EC-Earth3P-HR', 'CNRM-CM6-1',
                          'KIOST-ESM', 'FGOALS-f3-H', 'NorESM1-F',
                          'GISS-E2-1-G', 'IPSL-CM5A2-INCA', 'IPSL-CM6A-LR',
                          'INM-CM5-H', 'NorESM2-MM', 'CIESM', 'CESM1-WACCM-SC',
                          'SAM0-UNICON', 'HadGEM3-GC31-MM', 'ssp585', 'MIROC6',
                          'IPSL-CM6A-ATM-HR', 'ACCESS-ESM1-5',
                          'CESM2-WACCM-FV2', 'MPI-ESM-1-2-HAM', 'GFDL-CM4',
                          'HadGEM3-GC31-LM', 'EC-Earth3P', 'MCM-UA-1-0',
                          'GFDL-OM4p5B', 'GFDL-ESM2M', 'EC-Earth3-LR',
                          'EC-Earth3-Veg-LR', 'BCC-CSM2-HR', 'GFDL-AM4',
                          'FGOALS-f3-L', 'E3SM-1-0', 'CMCC-ESM2', 'E3SM-1-1',
                          'KACE-1-0-G', 'IITM-ESM', 'IPSL-CM6A-LR-INCA',
                          'MIROC-ES2L', 'GISS-E2-2-G', 'INM-CM5-0',
                          'NorESM2-LM', 'MRI-ESM2-0']},
 'table_id': {'count': 38,
              'values': ['EdayZ', 'AERmon', 'CF3hr', 'hus', 'IfxGre', 'Lmon',
                         '3hr', 'CFmon', 'Efx', 'SIclim', '6hrPlevPt', 'Eclim',
                         'Emon', 'Omon', 'Ofx', 'Odec', 'E1hrClimMon', 'fx',
                         'ImonGre', 'AERmonZ', 'Amon', 'AERday', 'day', 'CFday',
                         'Eday', 'Oclim', '6hrLev', 'Eyr', 'Aclim', 'E3hr',
                         'SImon', 'AERhr', 'LImon', 'SIday', '6hrPlev', 'Oday',
                         'Oyr', 'EmonZ']}}
```

Searching for specific datasets

In the example below, we are are going to search for the following:

- variables: `o2` which stands for `mole_concentration_of_dissolved_molecular_oxygen_in_sea_water`
- experiments: `['historical', 'ssp585']`:
  - `historical`: all forcing of the recent past.
  - `ssp585`: emission-driven RCP8.5 based on SSP5.
- table_id: `Oyr` which stands for annual mean variables on the ocean grid.

- grid_label: `gn` which stands for data reported on a model's native grid.

For more details on the CMIP6 vocabulary, please check this website, and Core Controlled Vocabularies (CVs) for use in CMIP6 GitHub repository.

```
cat = col.search(
    experiment_id=["historical", "ssp585"],
    table_id="Oyr",
    variable_id="o2",
    grid_label="gn",
)

cat
```

```
<IPython.core.display.HTML object>
```

```
cat.df.head()
```

```
  activity_id institution_id     source_id experiment_id  member_id table_id  \
0        CMIP            IPSL  IPSL-CM6A-LR    historical  r12i1p1f1      Oyr
1        CMIP            IPSL  IPSL-CM6A-LR    historical  r21i1p1f1      Oyr
2        CMIP            IPSL  IPSL-CM6A-LR    historical  r11i1p1f1      Oyr
3        CMIP            IPSL  IPSL-CM6A-LR    historical  r10i1p1f1      Oyr
4        CMIP            IPSL  IPSL-CM6A-LR    historical   r1i1p1f1      Oyr

  variable_id grid_label                                              zstore  \
0          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...
1          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...
2          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...
3          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...
4          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...

   dcpp_init_year   version
0             NaN  20180803
1             NaN  20180803
2             NaN  20180803
3             NaN  20180803
4             NaN  20180803
```

Loading datasets Using `to_dataset_dict()`

```
dset_dict = cat.to_dataset_dict(
    zarr_kwargs={"consolidated": True, "decode_times": True, "use_cftime": True}
)
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'activity_id.institution_id.source_id.experiment_id.table_id.grid_label'
```

```
<IPython.core.display.HTML object>
```

```
[key for key in dset_dict.keys()]
```

```
['ScenarioMIP.NCC.NorESM2-MM.ssp585.Oyr.gn',
 'CMIP.MRI.MRI-ESM2-0.historical.Oyr.gn',
```

(continues on next page)

(continued from previous page)

```
 'ScenarioMIP.IPSL.IPSL-CM6A-LR.ssp585.Oyr.gn',
 'ScenarioMIP.CMCC.CMCC-ESM2.ssp585.Oyr.gn',
 'ScenarioMIP.EC-Earth-Consortium.EC-Earth3-CC.ssp585.Oyr.gn',
 'ScenarioMIP.MIROC.MIROC-ES2L.ssp585.Oyr.gn',
 'CMIP.MPI-M.MPI-ESM1-2-HR.historical.Oyr.gn',
 'ScenarioMIP.DWD.MPI-ESM1-2-HR.ssp585.Oyr.gn',
 'CMIP.CCCma.CanESM5-CanOE.historical.Oyr.gn',
 'ScenarioMIP.NCAR.CESM2.ssp585.Oyr.gn',
 'ScenarioMIP.NCC.NorESM2-LM.ssp585.Oyr.gn',
 'CMIP.CCCma.CanESM5.historical.Oyr.gn',
 'CMIP.EC-Earth-Consortium.EC-Earth3-CC.historical.Oyr.gn',
 'CMIP.NCC.NorESM2-MM.historical.Oyr.gn',
 'ScenarioMIP.MPI-M.MPI-ESM1-2-LR.ssp585.Oyr.gn',
 'CMIP.CMCC.CMCC-ESM2.historical.Oyr.gn',
 'CMIP.CSIRO.ACCESS-ESM1-5.historical.Oyr.gn',
 'CMIP.MIROC.MIROC-ES2L.historical.Oyr.gn',
 'CMIP.HAMMOZ-Consortium.MPI-ESM-1-2-HAM.historical.Oyr.gn',
 'ScenarioMIP.CSIRO.ACCESS-ESM1-5.ssp585.Oyr.gn',
 'CMIP.MPI-M.MPI-ESM1-2-LR.historical.Oyr.gn',
 'CMIP.IPSL.IPSL-CM5A2-INCA.historical.Oyr.gn',
 'ScenarioMIP.CCCma.CanESM5.ssp585.Oyr.gn',
 'ScenarioMIP.DKRZ.MPI-ESM1-2-HR.ssp585.Oyr.gn',
 'CMIP.NCC.NorESM2-LM.historical.Oyr.gn',
 'CMIP.IPSL.IPSL-CM6A-LR.historical.Oyr.gn',
 'ScenarioMIP.MRI.MRI-ESM2-0.ssp585.Oyr.gn',
 'ScenarioMIP.CCCma.CanESM5-CanOE.ssp585.Oyr.gn']
```

We can access a particular dataset as follows:

```
ds = dset_dict["CMIP.CCCma.CanESM5.historical.Oyr.gn"]
print(ds)
```

```
<xarray.Dataset>
Dimensions:              (i: 360, j: 291, lev: 45, bnds: 2, member_id: 35, time: 165,␣
↪vertices: 4)
Coordinates:
  * i                    (i) int32 0 1 2 3 4 5 6 ... 353 354 355 356 357 358 359
  * j                    (j) int32 0 1 2 3 4 5 6 ... 284 285 286 287 288 289 290
    latitude             (j, i) float64 dask.array<chunksize=(291, 360), meta=np.
↪ndarray>
  * lev                  (lev) float64 3.047 9.454 16.36 ... 5.375e+03 5.625e+03
    lev_bnds             (lev, bnds) float64 dask.array<chunksize=(45, 2), meta=np.
↪ndarray>
    longitude            (j, i) float64 dask.array<chunksize=(291, 360), meta=np.
↪ndarray>
  * time                 (time) object 1850-07-02 12:00:00 ... 2014-07-02 12:0...
    time_bnds            (time, bnds) object dask.array<chunksize=(165, 2), meta=np.
↪ndarray>
  * member_id            (member_id) <U9 'r24i1p1f1' 'r16i1p1f1' ... 'r10i1p1f1'
Dimensions without coordinates: bnds, vertices
Data variables:
    o2                   (member_id, time, lev, j, i) float32 dask.array<chunksize=(1,␣
↪12, 45, 291, 360), meta=np.ndarray>
    vertices_latitude    (j, i, vertices) float64 dask.array<chunksize=(291, 360, 4),␣
↪meta=np.ndarray>
    vertices_longitude   (j, i, vertices) float64 dask.array<chunksize=(291, 360, 4),␣
↪meta=np.ndarray>
```

(continues on next page)

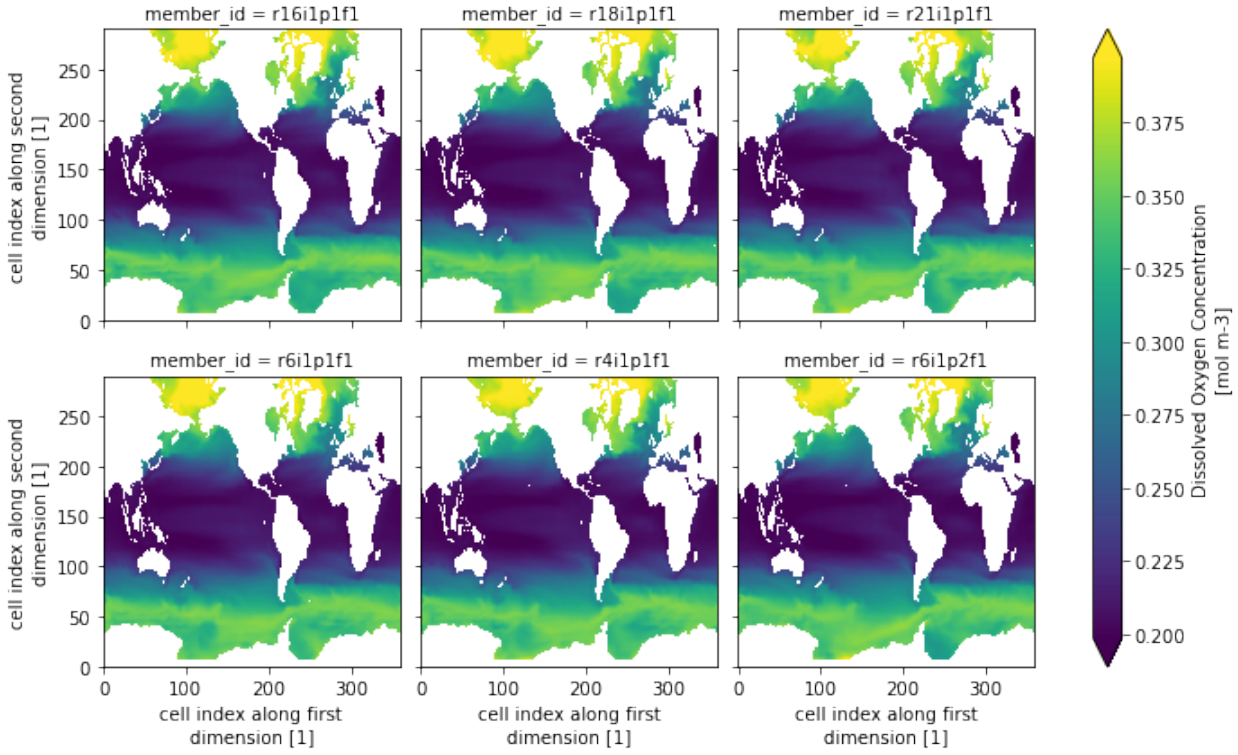
(continued from previous page)

```
Attributes: (12/58)
    source_id:                  CanESM5
    branch_time_in_child:       0.0
    contact:                    ec.cccma.info-info.ccmac.ec@canada.ca
    parent_activity_id:         CMIP
    CCCma_runid:                rc3.1-his10
    references:                 Geophysical Model Development Special issue ...
    ...                         ...
    table_info:                 Creation Date:(20 February 2019) MD5:374fbe5...
    CCCma_pycmor_hash:          33c30511acc319a98240633965a04ca99c26427e
    status:                     2019-10-25;created;by nhn2@columbia.edu
    parent_mip_era:             CMIP6
    sub_experiment_id:          none
    intake_esm_dataset_key:     CMIP.CCCma.CanESM5.historical.Oyr.gn
```

Let's create a quick plot for a slice of the data:

```
ds.o2.isel(time=0, lev=0, member_id=range(1, 24, 4)).plot(col="member_id", col_wrap=3,
↪ robust=True)
```

```
<xarray.plot.facetgrid.FacetGrid at 0x7efd93bfb610>
```



Using custom preprocessing functions

When comparing many models it is often necessary to preprocess (e.g. rename certain variables) them before running some analysis step. The `preprocess` argument lets the user pass a function, which is executed for each loaded asset before aggregations.

```
cat_pp = col.search(
    experiment_id=["historical"],
    table_id="Oyr",
    variable_id="o2",
    grid_label="gn",
    source_id=["IPSL-CM6A-LR", "CanESM5"],
    member_id="r10i1p1f1",
)
cat_pp.df
```

```
  activity_id institution_id     source_id experiment_id  member_id table_id  \
0        CMIP            IPSL  IPSL-CM6A-LR    historical  r10i1p1f1      Oyr
1        CMIP           CCCma       CanESM5    historical  r10i1p1f1      Oyr

  variable_id grid_label                                              zstore  \
0          o2         gn  gs://cmip6/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/histor...
1          o2         gn  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...

   dcpp_init_year   version
0             NaN  20180803
1             NaN  20190429
```

```
# load the example
dset_dict_raw = cat_pp.to_dataset_dict(zarr_kwargs={"consolidated": True})
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'activity_id.institution_id.source_id.experiment_id.table_id.grid_label'
```

```
<IPython.core.display.HTML object>
```

```
for k, ds in dset_dict_raw.items():
    print(f"dataset key={k}\n\tdimensions={sorted(list(ds.dims))}\n")
```

```
dataset key=CMIP.IPSL.IPSL-CM6A-LR.historical.Oyr.gn
        dimensions=['axis_nbounds', 'member_id', 'nvertex', 'olevel', 'time', 'x', 'y
↪']

dataset key=CMIP.CCCma.CanESM5.historical.Oyr.gn
        dimensions=['bnds', 'i', 'j', 'lev', 'member_id', 'time', 'vertices']
```

Note: Note that both models follow a different naming scheme. We can define a little helper function and pass it to `.to_dataset_dict()` to fix this. For demonstration purposes we will focus on the vertical level dimension which is called `lev` in `CanESM5` and `olevel` in `IPSL-CM6A-LR`.

```
def helper_func(ds):
    """Rename `olevel` dim to `lev`"""
    ds = ds.copy()
```

(continues on next page)

(continued from previous page)

```
    # a short example
    if "olevel" in ds.dims:
        ds = ds.rename({"olevel": "lev"})
    return ds
```

```
dset_dict_fixed = cat_pp.to_dataset_dict(zarr_kwargs={"consolidated": True},␣
↪preprocess=helper_func)
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'activity_id.institution_id.source_id.experiment_id.table_id.grid_label'
```

```
<IPython.core.display.HTML object>
```

```
for k, ds in dset_dict_fixed.items():
    print(f"dataset key={k}\n\tdimensions={sorted(list(ds.dims))}\n")
```

```
dataset key=CMIP.IPSL.IPSL-CM6A-LR.historical.Oyr.gn
        dimensions=['axis_nbounds', 'lev', 'member_id', 'nvertex', 'time', 'x', 'y']

dataset key=CMIP.CCCma.CanESM5.historical.Oyr.gn
        dimensions=['bnds', 'i', 'j', 'lev', 'member_id', 'time', 'vertices']
```

This was just an example for one dimension.

Note: Check out cmip6-preprocessing package for a full renaming function for all available CMIP6 models and some other utilities.

```
import intake_esm  # just to display version information

intake_esm.show_versions()
```

```
INSTALLED VERSIONS
------------------

cftime: 1.5.0
dask: 2021.08.0
fastprogress: 0.2.7
fsspec: 2021.07.0
gcsfs: 2021.07.0
intake: 0.6.3
intake_esm: 0.0.0
netCDF4: 1.5.7
pandas: 1.3.2
requests: 2.26.0
s3fs: 2021.07.0
xarray: 0.19.0
zarr: 2.8.3
```

5.2.5 Manipulating DataFrame (in-memory catalog)

```
import warnings

warnings.filterwarnings("ignore")
import intake
```

The in-memory representation of an Earth System Model (ESM) catalog is a pandas dataframe, and is accessible via the `.df` property:

```
url = "https://storage.googleapis.com/cmip6/pangeo-cmip6.json"
col = intake.open_esm_datastore(url)
col.df.head()
```

```
  activity_id institution_id     source_id       experiment_id member_id  \
0  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
1  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
2  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
3  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1
4  HighResMIP           CMCC  CMCC-CM2-HR4  highresSST-present  r1i1p1f1

  table_id variable_id grid_label  \
0     Amon          hus         gn
1     Amon         rsdt         gn
2     Amon          prw         gn
3     Amon         rlus         gn
4     Amon         rlds         gn

                                          zstore  dcpp_init_year   version
0  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
1  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
2  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
3  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
4  gs://cmip6/CMIP6/HighResMIP/CMCC/CMCC-CM2-HR4/...             NaN  20170706
```

In this notebook we will go through some examples showing how to manipulate this dataframe outside of intake-esm.

Use Case 1: Complex Search Queries

Let's say we are interested in datasets with the following attributes:

- `experiment_id=["historical"]`
- `table_id="Amon"`
- `variable_id="tas"`
- `source_id=['TaiESM1', 'AWI-CM-1-1-MR', 'AWI-ESM-1-1-LR', 'BCC-CSM2-MR', 'BCC-ESM1', 'CAMS-CSM1-0', 'CAS-ESM2-0', 'UKESM1-0-LL']`

In addition to these attributes, **we are interested in the first ensemble member (member_id) of each model (source_id) only**.

This can be achieved in two steps:

Step 1: Run a query against the catalog

We can run a query against the catalog:

```
col_subset = col.search(
    experiment_id=["historical"],
    table_id="Amon",
    variable_id="tas",
    source_id=[
        "TaiESM1",
        "AWI-CM-1-1-MR",
        "AWI-ESM-1-1-LR",
        "BCC-CSM2-MR",
        "BCC-ESM1",
        "CAMS-CSM1-0",
        "CAS-ESM2-0",
        "UKESM1-0-LL",
    ],
)
col_subset
```

```
<IPython.core.display.HTML object>
```

Step 2: Select the first `member_id` for each `source_id`

The subsetted catalog contains `source_id` with the following number of `member_id` per `source_id`:

```
col_subset.df.groupby("source_id")["member_id"].nunique()
```

```
source_id
AWI-CM-1-1-MR      5
AWI-ESM-1-1-LR     1
BCC-CSM2-MR        3
BCC-ESM1           3
CAMS-CSM1-0        3
CAS-ESM2-0         4
TaiESM1            2
UKESM1-0-LL       19
Name: member_id, dtype: int64
```

To get the first `member_id` for each `source_id`, we group the dataframe by `source_id` and use the `.first()` function to retrieve the first `member_id`:

```
grouped = col_subset.df.groupby(["source_id"])
df = grouped.first().reset_index()

# Confirm that we have one ensemble member per source_id

df.groupby("source_id")["member_id"].nunique()
```

```
source_id
AWI-CM-1-1-MR     1
AWI-ESM-1-1-LR    1
BCC-CSM2-MR       1
```

(continues on next page)

(continued from previous page)

```
BCC-ESM1          1
CAMS-CSM1-0       1
CAS-ESM2-0        1
TaiESM1           1
UKESM1-0-LL       1
Name: member_id, dtype: int64
```

Step 3: Attach the new dataframe to our catalog object

```
col_subset.df = df
col_subset
```

```
<IPython.core.display.HTML object>
```

```
dsets = col_subset.to_dataset_dict(zarr_kwargs={"consolidated": True})
[key for key in dsets]
```

```
--> The keys in the returned dictionary of datasets are constructed as follows:
        'activity_id.institution_id.source_id.experiment_id.table_id.grid_label'
```

```
<IPython.core.display.HTML object>
```

```
['CMIP.BCC.BCC-CSM2-MR.historical.Amon.gn',
 'CMIP.AS-RCEC.TaiESM1.historical.Amon.gn',
 'CMIP.BCC.BCC-ESM1.historical.Amon.gn',
 'CMIP.CAMS.CAMS-CSM1-0.historical.Amon.gn',
 'CMIP.AWI.AWI-CM-1-1-MR.historical.Amon.gn',
 'CMIP.MOHC.UKESM1-0-LL.historical.Amon.gn',
 'CMIP.AWI.AWI-ESM-1-1-LR.historical.Amon.gn',
 'CMIP.CAS.CAS-ESM2-0.historical.Amon.gn']
```

```
print(dsets["CMIP.CAS.CAS-ESM2-0.historical.Amon.gn"])
```

```
<xarray.Dataset>
Dimensions:    (lat: 128, bnds: 2, lon: 256, member_id: 1, time: 1980)
Coordinates:
    height     float64 ...
  * lat        (lat) float64 -90.0 -88.58 -87.17 -85.75 ... 87.17 88.58 90.0
    lat_bnds   (lat, bnds) float64 dask.array<chunksize=(128, 2), meta=np.ndarray>
  * lon        (lon) float64 0.0 1.406 2.812 4.219 ... 354.4 355.8 357.2 358.6
    lon_bnds   (lon, bnds) float64 dask.array<chunksize=(256, 2), meta=np.ndarray>
  * time       (time) object 1850-01-16 12:00:00 ... 2014-12-16 12:00:00
    time_bnds  (time, bnds) object dask.array<chunksize=(1980, 2), meta=np.ndarray>
  * member_id  (member_id) <U8 'r1i1p1f1'
Dimensions without coordinates: bnds
Data variables:
    tas        (member_id, time, lat, lon) float32 dask.array<chunksize=(1, 381, 128,␣
↪256), meta=np.ndarray>
Attributes: (12/51)
    Conventions:            CF-1.7 CMIP-6.2
    activity_id:            CMIP
    branch_method:          standard
```

(continues on next page)

(continued from previous page)

```
    branch_time_in_child:     0.0
    branch_time_in_parent:    0.0
    cmor_version:             3.5.0
    ...                       ...
    variable_id:              tas
    variant_label:            r1i1p1f1
    netcdf_tracking_ids:      hdl:21.14100/22e89a1b-f73e-45be-84dc-7d0aabbeea9d
    version_id:               v20200302
    intake_esm_varname:       ['tas']
    intake_esm_dataset_key:   CMIP.CAS.CAS-ESM2-0.historical.Amon.gn
```

```
import intake_esm  # just to display version information

intake_esm.show_versions()
```

```
INSTALLED VERSIONS
------------------

cftime: 1.5.0
dask: 2021.08.0
fastprogress: 0.2.7
fsspec: 2021.07.0
gcsfs: 2021.07.0
intake: 0.6.3
intake_esm: 0.0.0
netCDF4: 1.5.7
pandas: 1.3.2
requests: 2.26.0
s3fs: 2021.07.0
xarray: 0.19.0
zarr: 2.8.3
```

5.3 Supplemental Guide

5.3.1 Frequently Asked Questions

How do I create my own catalog?

Intake-esm catalogs include two pieces:

1. **An ESM-Collection file**: an ESM-Collection file is a simple json file that provides metadata about the catalog. The specification for this json file is found in the esm-collection-spec repository.
2. **A catalog file**: the catalog file is a CSV file that lists the catalog contents. This file includes one row per dataset granule (e.g. a NetCDF file or Zarr dataset). The columns in this CSV must match the attributes and assets listed in the ESM-Collection file. A short example of a catalog file is shown below::

```
activity_id,institution_id,source_id,experiment_id,member_id,table_id,variable_id,
↪grid_label,zstore,dcpp_init_year
AerChemMIP,BCC,BCC-ESM1,piClim-CH4,r1i1p1f1,Amon,ch4,gn,gs://cmip6/AerChemMIP/BCC/
↪BCC-ESM1/piClim-CH4/r1i1p1f1/Amon/ch4/gn/,
AerChemMIP,BCC,BCC-ESM1,piClim-CH4,r1i1p1f1,Amon,clt,gn,gs://cmip6/AerChemMIP/BCC/
↪BCC-ESM1/piClim-CH4/r1i1p1f1/Amon/clt/gn/,
```

(continues on next page)

(continued from previous page)

```
AerChemMIP,BCC,BCC-ESM1,piClim-CH4,r1i1p1f1,Amon,co2,gn,gs://cmip6/AerChemMIP/BCC/
↪BCC-ESM1/piClim-CH4/r1i1p1f1/Amon/co2/gn/,
AerChemMIP,BCC,BCC-ESM1,piClim-CH4,r1i1p1f1,Amon,evspsbl,gn,gs://cmip6/AerChemMIP/
↪BCC/BCC-ESM1/piClim-CH4/r1i1p1f1/Amon/evspsbl/gn/,
...
```

Is there a list of existing catalogs?

The table below is an incomplete list of existing catalogs. Please feel free to add to this list or raise an issue on GitHub.

CMIP6-GLADE

- *Description*: **CMIP6 data accessible on the NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cmip6.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/CMIP6/Guide/dataUsers.html

CMIP6-CESM2-Timeseries

- *Description*: **CESM2 raw output (non-cmorized) that went into CMIP6 data**
- *Platform*: **NCAR-CAMPAIGN**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/campaign-cesm2-cmip6-timeseries.json**
- *Data Format*: **netCDF**
- Documentation Page: http://www.cesm.ucar.edu/models/cesm2/

CMIP5-GLADE

- *Description*: **CMIP5 data accessible on the NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cmip5.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/mips/cmip5/guide.html

CESM1-LENS-AWS

- *Description*: **CESM1 Large Ensemble data publicly available on Amazon S3**
- *Platform*: **AWS S3 (us-west-2 region)**
- *Catalog path or url*: **https://raw.githubusercontent.com/NCAR/cesm-lens-aws/master/intake-catalogs/aws-cesm1-le.json**
- *Data Format*: **Zarr**
- Documentation Page: https://doi.org/10.26024/wt24-5j82

CESM1-LENS-GLADE

- *Description*: **CESM1 Large Ensemble data stored on NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**

- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cesm1-le.json**
- *Data Format*: **netCDF**
- Documentation Page: https://doi.org/10.5065/d6j101d1

CESM2-LE-GLADE

- *Description*: **ESM collection for the CESM2 LENS data stored on GLADE in /glade/campaign/cgd/cesm/CESM2-LE/timeseries**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cesm2-le.json**
- *Data Format*: **netCDF**
- Documentation Page: https://www.cesm.ucar.edu/projects/community-projects/LENS2/

CMIP6-GCP

- *Description*: **CMIP6 Zarr data residing in Pangeo's Google Storage**
- *Platform*: **Google Cloud Platform**
- *Catalog path or url*: **https://storage.googleapis.com/cmip6/pangeo-cmip6.json**
- *Data Format*: **Zarr**
- Documentation Page: https://pcmdi.llnl.gov/CMIP6/Guide/dataUsers.html

CMIP6-MISTRAL

- *Description*: **CMIP6 data accessible on the DKRZ's MISTRAL disk storage system**
- *Platform*: **DKRZ (German Climate Computing Centre)-MISTRAL**
- *Catalog path or url*: **/work/ik1017/Catalogs/mistral-cmip6.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/CMIP6/Guide/dataUsers.html

CMIP5-MISTRAL

- *Description*: **CMIP5 data accessible on the DKRZ's MISTRAL disk storage system**
- *Platform*: **DKRZ (German Climate Computing Centre)-MISTRAL**
- *Catalog path or url*: **/work/ik1017/Catalogs/mistral-cmip5.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/mips/cmip5/guide.html

MiKlip-MISTRAL

- *Description*: **Data from MiKlip projects at the Max Planck Institute for Meteorology (MPI-M)**
- *Platform*: **DKRZ (German Climate Computing Centre)-MISTRAL**
- *Catalog path or url*: **/work/ik1017/Catalogs/mistral-miklip.json**
- *Data Format*: **netCDF**
- Documentation Page: https://www.fona-miklip.de/

MPI-GE-MISTRAL

- *Description*: **Max Planck Institute Grand Ensemble cmorized by CMIP5-standards**

- *Platform*: **DKRZ (German Climate Computing Centre)-MISTRAL**
- *Catalog path or url*: **/work/ik1017/Catalogs/mistral-MPI-GE.json**
- *Data Format*: **netCDF**
- Documentation Page: https://doi.org/10/gf3kgt

CMIP6-LDEO-OpenDAP

- *Description*: **CMIP6 data accessible via Hyrax OpenDAP Server at Lamont-Doherty Earth Observatory**
- *Platform*: **LDEO-OpenDAP**
- *Catalog path or url*: **http://haden.ldeo.columbia.edu/catalogs/hyrax_cmip6.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/CMIP6/Guide/dataUsers.html

---

Note

Some of these catalogs are also stored in intake-esm-datastore GitHub repository at https://github.com/NCAR/intake-esm-datastore/tree/master/catalogs

---

5.3.2 NCAR CMIP Analysis Platform

NCAR's CMIP Analysis Platform (CMIP AP) includes a large collection of CMIP5 and CMIP6 data sets.

Requesting data sets

Use this form to request new data be added to the CMIP AP. Typically requests are fulfilled within two weeks. Contact CISL if you have further questions. Intake-ESM catalogs are regularly updated following the addition (or removal) of data from the platform.

Available catalogs at NCAR

NCAR has created multiple Intake ESM catalogs that work on datasets stored on GLADE. Those catalogs are listed below:

CMIP6-GLADE

- *Description*: **CMIP6 data accessible on the NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cmip6.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/CMIP6/Guide/dataUsers.html

CMIP6-CESM2-Timeseries

- *Description*: **CESM2 raw output (non-cmorized) that went into CMIP6 data**
- *Platform*: **NCAR-CAMPAIGN**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/campaign-cesm2-cmip6-timeseries.json**

- *Data Format*: **netCDF**
- Documentation Page: http://www.cesm.ucar.edu/models/cesm2/

CMIP5-GLADE

- *Description*: **CMIP5 data accessible on the NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cmip5.json**
- *Data Format*: **netCDF**
- Documentation Page: https://pcmdi.llnl.gov/mips/cmip5/guide.html

CESM1-LENS-GLADE

- *Description*: **CESM1 Large Ensemble data stored on NCAR's GLADE disk storage system**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cesm1-le.json**
- *Data Format*: **netCDF**
- Documentation Page: https://doi.org/10.5065/d6j101d1

CESM2-LE-GLADE

- *Description*: **ESM collection for the CESM2 LENS data stored on GLADE in /glade/campaign/cgd/cesm/CESM2-LE/timeseries**
- *Platform*: **NCAR-GLADE**
- *Catalog path or url*: **/glade/collections/cmip/catalog/intake-esm-datastore/catalogs/glade-cesm2-le.json**
- *Data Format*: **netCDF**
- Documentation Page: https://www.cesm.ucar.edu/projects/community-projects/LENS2/

5.4 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.

5.4.1 ESM Datastore (`intake.open_esm_datastore`)

class intake_esm.core.**esm_datastore**(**args*, ***kwargs*)

An intake plugin for parsing an ESM (Earth System Model) Collection/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

- **esmcol_obj** (`str`, `pandas.DataFrame`) – If string, this must be a path or URL to an ESM collection JSON file. If pandas.DataFrame, this must be the catalog content that would otherwise be in a CSV file.
- **esmcol_data** (`dict`, *optional*) – ESM collection spec information, by default None
- **progressbar** (`bool`, *optional*) – Will print a progress bar to standard error (stderr) when loading assets into `Dataset`, by default True

- **sep** (`str`, *optional*) – Delimiter to use when constructing a key for a query, by default '.'
- **csv_kwargs** (`dict`, *optional*) – Additional keyword arguments passed through to the `read_csv()` function.
- ****kwargs** – 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"
>>> col = intake.open_esm_datastore(url)
>>> col.df.head()
activity_id institution_id source_id experiment_id  ... variable_id grid_label
↪                                        zstore dcpp_init_year
0  AerChemMIP            BCC  BCC-ESM1        ssp370  ...          pr         gn
↪gs://cmip6/AerChemMIP/BCC/BCC-ESM1/ssp370/r1i1...            NaN
1  AerChemMIP            BCC  BCC-ESM1        ssp370  ...        prsn         gn
↪gs://cmip6/AerChemMIP/BCC/BCC-ESM1/ssp370/r1i1...            NaN
2  AerChemMIP            BCC  BCC-ESM1        ssp370  ...         tas         gn
↪gs://cmip6/AerChemMIP/BCC/BCC-ESM1/ssp370/r1i1...            NaN
3  AerChemMIP            BCC  BCC-ESM1        ssp370  ...      tasmax         gn
↪gs://cmip6/AerChemMIP/BCC/BCC-ESM1/ssp370/r1i1...            NaN
4  AerChemMIP            BCC  BCC-ESM1        ssp370  ...      tasmin         gn
↪gs://cmip6/AerChemMIP/BCC/BCC-ESM1/ssp370/r1i1...            NaN
```

classmethod from_df(*df*, *esmcol_data=None*, *progressbar=True*, *sep='.'*, ***kwargs*)

Create catalog from the given dataframe

Parameters

- **df** (`pandas.DataFrame`) – catalog content that would otherwise be in a CSV file.
- **esmcol_data** (`dict`, *optional*) – ESM collection spec information, by default None
- **progressbar** (`bool`, *optional*) – Will print a progress bar to standard error (stderr) when loading assets into `Dataset`, by default True
- **sep** (`str`, *optional*) – Delimiter to use when constructing a key for a query, by default '.'

Returns *esm_datastore* – Catalog object

keys()

Get keys for the catalog entries

Returns `list` – keys for the catalog entries

nunique()

Count distinct observations across dataframe columns in the catalog.

Examples

```
>>> import intake
>>> col = intake.open_esm_datastore("pangeo-cmip6.json")
>>> col.nunique()
activity_id          10
institution_id       23
source_id            48
experiment_id        29
member_id            86
table_id             19
variable_id         187
grid_label            7
zstore            27437
dcpp_init_year       59
dtype: int64
```

search (*require_all_on=None*, ***query*)
Search for entries in the catalog.

Parameters

- **require_all_on** (`list`, `str`, *optional*) – A dataframe column or a list of dataframe columns across which all entries must satisfy the query criteria. If None, return entries that fulfill any of the criteria specified in the query, by default None.
- ****query** – keyword arguments corresponding to user's query to execute against the dataframe.

Returns **cat** (*esm_datastore*) – A new Catalog with a subset of the entries in this Catalog.

Examples

```
>>> import intake
>>> col = intake.open_esm_datastore("pangeo-cmip6.json")
>>> col.df.head(3)
activity_id institution_id source_id  ... grid_label
↪                   zstore dcpp_init_year
0  AerChemMIP            BCC  BCC-ESM1  ...         gn  gs://cmip6/AerChemMIP/
↪BCC/BCC-ESM1/ssp370/r1i1...             NaN
1  AerChemMIP            BCC  BCC-ESM1  ...         gn  gs://cmip6/AerChemMIP/
↪BCC/BCC-ESM1/ssp370/r1i1...             NaN
2  AerChemMIP            BCC  BCC-ESM1  ...         gn  gs://cmip6/AerChemMIP/
↪BCC/BCC-ESM1/ssp370/r1i1...             NaN
```

```
>>> cat = col.search(
...     source_id=["BCC-CSM2-MR", "CNRM-CM6-1", "CNRM-ESM2-1"],
...     experiment_id=["historical", "ssp585"],
...     variable_id="pr",
...     table_id="Amon",
...     grid_label="gn",
... )
>>> cat.df.head(3)
    activity_id institution_id    source_id  ... grid_label
↪                         zstore dcpp_init_year
260        CMIP            BCC  BCC-CSM2-MR  ...         gn  gs://cmip6/CMIP/
↪BCC/BCC-CSM2-MR/historical/r1i...           NaN
```

(continues on next page)

(continued from previous page)

```
346        CMIP            BCC  BCC-CSM2-MR  ...         gn  gs://cmip6/CMIP/
↪BCC/BCC-CSM2-MR/historical/r2i...            NaN
401        CMIP            BCC  BCC-CSM2-MR  ...         gn  gs://cmip6/CMIP/
↪BCC/BCC-CSM2-MR/historical/r3i...            NaN
```

The search method also accepts compiled regular expression objects from `compile()` as patterns.

```
>>> import re
>>> # Let's search for variables containing "Frac" in their name
>>> pat = re.compile(r"Frac")  # Define a regular expression
>>> cat.search(variable_id=pat)
>>> cat.df.head().variable_id
0     residualFrac
1    landCoverFrac
2    landCoverFrac
3     residualFrac
4    landCoverFrac
```

serialize(*name*, *directory=None*, *catalog_type='dict'*)

Serialize collection/catalog to corresponding json and csv files.

Parameters

- **name** (`str`) – name to use when creating ESM collection json file and csv catalog.
- **directory** (`str`, `PathLike`, *default* `None`) – The path to the local directory. If None, use the current directory
- **catalog_type** (`str`, *default* `'dict'`) – Whether to save the catalog table as a dictionary in the JSON file or as a separate CSV file.

Notes

Large catalogs can result in large JSON files. To keep the JSON file size manageable, call with *catalog_type='file'* to save catalog as a separate CSV file.

Examples

```
>>> import intake
>>> col = intake.open_esm_datastore("pangeo-cmip6.json")
>>> col_subset = col.search(
...     source_id="BCC-ESM1",
...     grid_label="gn",
...     table_id="Amon",
...     experiment_id="historical",
... )
>>> col_subset.serialize(name="cmip6_bcc_esm1", catalog_type="file")
Writing csv catalog to: cmip6_bcc_esm1.csv.gz
Writing ESM collection json file to: cmip6_bcc_esm1.json
```

to_dataset_dict(*zarr_kwargs=None*, *cdf_kwargs=None*, *preprocess=None*, *storage_options=None*, *progressbar=None*, *aggregate=None*)

Load catalog entries into a dictionary of xarray datasets.

Parameters

- **zarr_kwargs** (dict) – Keyword arguments to pass to open_zarr() function
- **cdf_kwargs** (dict) – Keyword arguments to pass to open_dataset() function. If specifying chunks, the chunking is applied to each netcdf file. Therefore, chunks must refer to dimensions that are present in each netcdf file, or chunking will fail.
- **preprocess** (callable, *optional*) – If provided, call this function on each dataset prior to aggregation.
- **storage_options** (dict, *optional*) – Parameters passed to the backend file-system such as Google Cloud Storage, Amazon Web Service S3.
- **progressbar** (bool) – If True, will print a progress bar to standard error (stderr) when loading assets into Dataset.
- **aggregate** (bool, *optional*) – If False, no aggregation will be done.

Returns **dsets** (dict) – A dictionary of xarray Dataset.

Examples

```
>>> import intake
>>> col = intake.open_esm_datastore("glade-cmip6.json")
>>> cat = col.search(
...     source_id=["BCC-CSM2-MR", "CNRM-CM6-1", "CNRM-ESM2-1"],
...     experiment_id=["historical", "ssp585"],
...     variable_id="pr",
...     table_id="Amon",
...     grid_label="gn",
... )
>>> dsets = cat.to_dataset_dict()
>>> dsets.keys()
dict_keys(['CMIP.BCC.BCC-CSM2-MR.historical.Amon.gn', 'ScenarioMIP.BCC.BCC-
↪CSM2-MR.ssp585.Amon.gn'])
>>> dsets["CMIP.BCC.BCC-CSM2-MR.historical.Amon.gn"]
<xarray.Dataset>
Dimensions:    (bnds: 2, lat: 160, lon: 320, member_id: 3, time: 1980)
Coordinates:
* lon         (lon) float64 0.0 1.125 2.25 3.375 ... 355.5 356.6 357.8 358.9
* lat         (lat) float64 -89.14 -88.03 -86.91 -85.79 ... 86.91 88.03 89.14
* time        (time) object 1850-01-16 12:00:00 ... 2014-12-16 12:00:00
* member_id   (member_id) <U8 'r1i1p1f1' 'r2i1p1f1' 'r3i1p1f1'
Dimensions without coordinates: bnds
Data variables:
    lat_bnds   (lat, bnds) float64 dask.array<chunksize=(160, 2), meta=np.
↪ndarray>
    lon_bnds   (lon, bnds) float64 dask.array<chunksize=(320, 2), meta=np.
↪ndarray>
    time_bnds  (time, bnds) object dask.array<chunksize=(1980, 2), meta=np.
↪ndarray>
    pr         (member_id, time, lat, lon) float32 dask.array<chunksize=(1,␣
↪600, 160, 320), meta=np.ndarray>
```

unique (*columns=None*)

Return unique values for given columns in the catalog.

Parameters **columns** (str, list) – name of columns for which to get unique values

Returns **info** (dict) – dictionary containing count, and unique values

Examples

```
>>> import intake
>>> import pprint
>>> col = intake.open_esm_datastore("pangeo-cmip6.json")
>>> uniques = col.unique(columns=["activity_id", "source_id"])
>>> pprint.pprint(uniques)
{'activity_id': {'count': 10,
                 'values': ['AerChemMIP',
                            'C4MIP',
                            'CMIP',
                            'DAMIP',
                            'DCPP',
                            'HighResMIP',
                            'LUMIP',
                            'OMIP',
                            'PMIP',
                            'ScenarioMIP']},
'source_id': {'count': 17,
              'values': ['BCC-ESM1',
                         'CNRM-ESM2-1',
                         'E3SM-1-0',
                         'MIROC6',
                         'HadGEM3-GC31-LL',
                         'MRI-ESM2-0',
                         'GISS-E2-1-G-CC',
                         'CESM2-WACCM',
                         'NorCPM1',
                         'GFDL-AM4',
                         'GFDL-CM4',
                         'NESM3',
                         'ECMWF-IFS-LR',
                         'IPSL-CM6A-ATM-HR',
                         'NICAM16-7S',
                         'GFDL-CM4C192',
                         'MPI-ESM1-2-HR']}}
```

update_aggregation(*attribute_name*, *agg_type=None*, *options=None*, *delete=False*)

Updates aggregation operations info.

Parameters

- **attribute_name** (`str`) – Name of attribute (column) across which to aggregate.
- **agg_type** (`str`, *optional*) – Type of aggregation operation to apply. Valid values include: *join_new*, *join_existing*, *union*, by default None
- **options** (`dict`, *optional*) – Aggregation settings that are passed as keywords arguments to `concat()` or `merge()`. For *join_existing*, it must contain the name of the existing dimension to use (for e.g.: something like {'dim': 'time'})., by default None
- **delete** (`bool`, *optional*) – Whether to delete/remove/disable aggregation operations for a particular attribute, by default False

property agg_columns

List of columns used to merge/concatenate compatible multiple `Dataset` into a single `Dataset`.

property data_format

The data format. Valid values are netcdf and zarr. If specified, it means that all data assets in the catalog use the same data format.

property df
Return pandas `DataFrame`.

property format_column_name
Name of the column which contains the data format.

property groupby_attrs
Dataframe columns used to determine groups of compatible datasets.

Returns `list` – Columns used to determine groups of compatible datasets.

property key_template
Return string template used to create catalog entry keys

Returns `str` – string template used to create catalog entry keys

property path_column_name
The name of the column containing the path to the asset.

property variable_column_name
Name of the column that contains the variable name.

5.5 Contribution Guide

Interested in helping build intake-esm? Have code from your work that you believe others will find useful? Have a few minutes to tackle an issue?

Contributions are highly welcomed and appreciated. Every little help counts, so do not hesitate!

The following sections cover some general guidelines regarding development in intake-esm for maintainers and contributors. Nothing here is set in stone and can't be changed. Feel free to suggest improvements or changes in the workflow.

5.5.1 Feature requests and feedback

We'd also like to hear about your propositions and suggestions. Feel free to submit them as issues on intake-esm's GitHub issue tracker and:

- Explain in detail how they should work.
- Keep the scope as narrow as possible. This will make it easier to implement.

5.5.2 Report bugs

Report bugs for intake-esm in the issue tracker.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting, specifically the Python interpreter version, installed libraries, and intake-esm version.
- Detailed steps to reproduce the bug.

If you can write a demonstration test that currently fails but should pass (xfail), that is a very useful commit to make as well, even if you cannot fix the bug itself.

5.5.3 Fix bugs

Look through the GitHub issues for bugs.

Talk to developers to find out how you can fix specific bugs.

5.5.4 Write documentation

intake-esm could always use more documentation. What exactly is needed?

- More complementary documentation. Have you perhaps found something unclear?
- Docstrings. There can never be too many of them.
- Blog posts, articles and such – they're all very appreciated.

You can also edit documentation files directly in the GitHub web interface, without using a local copy. This can be convenient for small fixes.

Build the documentation locally with the following command:

```
$ make docs
```

5.5.5 Preparing Pull Requests

1. Fork the intake-esm GitHub repository.
2. Clone your fork locally using git, connect your repository to the upstream (main project), and create a branch::

   ```
   $ git clone git@github.com:YOUR_GITHUB_USERNAME/intake-esm.git
   $ cd intake-esm
   $ git remote add upstream git@github.com:intake/intake-esm.git
   ```

   now, to fix a bug or add feature create your own branch off "master":

   ```
   $ git checkout -b your-bugfix-feature-branch-name master
   ```

   If you need some help with Git, follow this quick start guide: https://git.wiki.kernel.org/index.php/QuickStart
3. Install dependencies into a new conda environment::

```
$ conda env update -f ci/environment.yml
$ conda activate intake-esm-dev
```

4. Make an editable install of intake-esm by running::

```
$ python -m pip install -e .
```

5. Install `pre-commit <https://pre-commit.com>_` hooks on the intake-esm repo::

```
$ pre-commit install
```

   Afterwards `pre-commit` will run whenever you commit.

   pre-commit is a framework for managing and maintaining multi-language pre-commit hooks to ensure code-style and code formatting is consistent.

   Now you have an environment called `intake-esm-dev` that you can work in. You'll need to make sure to activate that environment next time you want to use it after closing the terminal or your system.

6. (Optional) Run all the tests

   Now running tests is as simple as issuing this command::

```
$ pytest --cov=./
```

   This command will run tests via the `pytest` tool.

7. Commit and push once your tests pass and you are happy with your change(s)::

   When committing, `pre-commit` will re-format the files if necessary.

```
$ git commit -a -m "<commit message>"
$ git push -u
```

8. Finally, submit a pull request through the GitHub website using this data::

```
head-fork: YOUR_GITHUB_USERNAME/intake-esm
compare: your-branch-name

base-fork: intake/intake-esm
base: master # if it's a bugfix or feature
```

5.6 Changelog

5.6.1 Intake-esm v2021.8.17

(full changelog)

Enhancements made

- Add pydantic models to facilitate data validation #347 (@andersy005)

Maintenance and upkeep improvements

- [pre-commit.ci] pre-commit autoupdate #355 (@pre-commit-ci)
- skip cmip6_preprocessing tests for the time being #354 (@andersy005)
- Bump styfle/cancel-workflow-action from 0.9.0 to 0.9.1 #348 (@dependabot)
- Update pre-commit hooks #346 (@andersy005)
- Bump codecov/codecov-action from 1 to 2.0.2 #345 (@dependabot)
- Disable workflows on Forks #342 (@andersy005)
- Add missing test dependency #340 (@andersy005)
- Code refactoring #338 (@andersy005)
- Bump pre-commit/action from v2.0.2 to v2.0.3 #337 (@dependabot)
- Bump styfle/cancel-workflow-action from 0.8.0 to 0.9.0 #334 (@dependabot)
- Bump pre-commit/action from v2.0.0 to v2.0.2 #333 (@dependabot)
- Bump styfle/cancel-workflow-action from 0.7.0 to 0.8.0 #322 (@dependabot)
- Fix CI #321 (@andersy005)
- Fix Tests: Use a publicly available s3 object #318 (@andersy005)
- Bump styfle/cancel-workflow-action from 0.6.0 to 0.7.0 #316 (@dependabot)

Documentation improvements

- Add cesm2-le catalog #349 (@mgrover1)
- Docs: Execute all notebooks #341 (@andersy005)
- Enable comments in docs via sphinx-comments #326 (@andersy005)

Other merged PRs

- pin pandas version #356 (@mgrover1)

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @dependabot | @mgrover1 | @pre-commit-ci

5.6.2 Intake-esm v2021.1.15

(full changelog)

Bug Fixes

- Fix memory error when computing unique values #313 (@andersy005)

Breaking Changes

- Drop support for Python 3.6 #311 (@andersy005)

Internal Changes

- Upgrade dependencies & pin versions in CI environment #314 (@andersy005)
- Fix failing upstream-dev CI #310 (@andersy005)

Documentation

- Update MPI catalogs for MISTRAL #308 (@aaronspring)

Contributors to this release

(GitHub contributors page for this release)

@aaronspring | @andersy005 | @jbusecke

5.6.3 Intake-esm v2020.12.18

(full changelog)

Bug Fixes

- Disable `_requested_variables` for single variable assets #306 (@andersy005)

Internal Changes

- Update changelog in preparation for new release #307 (@andersy005)
- Use `github-activity` to update list of contributors #302 (@andersy005)
- Add nbqa & Update prettier commit hooks #300 (@andersy005)
- Update pre-commit and GH actions #299 (@andersy005)

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @dcherian | @jbusecke | @naomi-henderson | @Recalculate

5.6.4 Intake-esm v2020.11.4

Features

- Support multiple variable assets/files. (GH#287) @andersy005
- Add utility function for printing version information. (GH#284) @andersy005

Breaking Changes

- Remove unnecessary logging bits. (GH#297) @andersy005

Bug Fixes

- ✓ Fix test failures. (GH#280) @andersy005
- Fix TypeError bug in `.search()` method when using wildcard and regular expressions. (GH#285) @andersy005
- Use file like object when dealing with netcdf in the cloud. (GH#292) @andersy005

Documentation

- Fix ReadtheDocs documentation builds. (GH#286) @andersy005
- Migrate docs from restructured text to markdown via `myst-parsers`. (GH#296) @andersy005
- Refactor documentation contents & add new notebooks. (GH#298) @andersy005

Internal Changes

- Fix import errors due to intake/intake#526. (GH#282) @andersy005
- Migrate CI from CircleCI to GitHub Actions. (GH#283) @andersy005
- Use mamba to speed up CI testing. (GH#293) @andersy005
- Enable dependabot updates. (GH#294) @andersy005
- Test against Python 3.9. (GH#295) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @dcherian | @jbusecke | @jukent | @sherimickelson

5.6.5 Intake-esm v2020.8.15

Features

- Support regular expression objects in `search()` (GH#236) @andersy005
- Support wildcard expresssions in `search()` (GH#259) @andersy005
- Expose attributes used when aggregating/combining datasets (GH#268) @andersy005
- Support turning aggregations off (GH#269) @andersy005
- Improve error messages (GH#270) @andersy005
- Expose aggregations options passed to xarray during datasets aggregation (GH#272) @andersy005
- Reset `_entries` dict after updating aggregations (GH#274) @andersy005

Documentation

- Update `to_dataset_dict()` docstring to inform users on how `cdf_kwargs` argument is used in regards to chunking (GH#278) @bonnland

Internal Changes

- Update pre-commit hooks & GitHub actions (GH#260) @andersy005
- Update badges (GH#258) @andersy005
- Update upstream environment (GH#263) @andersy005
- Refactor search functionality into a standalone module (GH#267) @andersy005
- Fix dask/concurrent.futures parallelism (GH#271) @andersy005
- Increase test coverage to ~100% (GH#273) @andersy005
- Bump minimum required versions (GH#275) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @bonnland | @dcherian | @jeffdlb | @jukent | @kmpaul | @markusritschel | @martindurant | @mattlong

5.6.6 Intake-esm v2020.6.11

Features

- Add `df` property setter (GH#247) @andersy005

Documentation

- Use Pandas sphinx theme (GH#244) @andersy005
- Update documentation tutorial (GH#252) @andersy005 & @charlesbluca

Internal Changes

- Fix anti-patterns and other bug risks (GH#251) @andersy005
- Sync with intake's Entry unification (GH#249) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @jhamman | @martindurant

5.6.7 Intake-esm v2020.5.21

Features

- Provide informative message/warnings from empty queries. (GH#235) @andersy005
- Replace tqdm progressbar with fastprogress. (GH#238) @andersy005
- Add `catalog_file` attribute to `esm_datastore` class. (GH#240) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @bonnland | @dcherian | @jbusecke | @jeffdlb | @kmpaul | @markusritschel

5.6.8 Intake-esm v2020.5.01

Features

- Add html representation for the catalog object. (GH#229) @andersy005
- Move logic for assets aggregation into `ESMGroupDataSource()` and add few basic dict-like methods (`keys()`, `len()`, `getitem()`, `contains()`) to the catalog object. (GH#194) @andersy005 & @jhamman & @kmpaul
- Support columns with iterables in *`unique()`* and *`nunique()`*. (GH#223) @andersy005

Bug Fixes

- Revert back to using `concurrent.futures` to address failures due to dask's distributed scheduler. (GH#225) & (GH#226)

Internal Changes

- Increase test coverage. (GH#222) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @bonnland | @dcherian | @jbusecke | @jhamman | @kmpaul | @sherimickelson

5.6.9 Intake-esm v2020.3.16

Features

- Support single file catalogs. (GH#195) @bonnland
- Add `progressbar` argument to *`to_dataset_dict()`*. This allows the user to override the default `progressbar` value used during the class instantiation. (GH#204) @andersy005
- Enhanced search: enforce query criteria via `require_all_on` argument via *`search()`* method. (GH#202) & (GH#207) & (GH#209) @andersy005 & @jbusecke
- Support relative paths for catalog files. (GH#208) @andersy005

Bug Fixes

- Use raw path if protocol is `None`. (GH#210) @andersy005

Internal Changes

- Github Action to publish package to PyPI on release. (GH#190) @andersy005
- Remove unnecessary inheritance. (GH#193) @andersy005
- Update linting GitHub action to run on all pull requests. (GH#196) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @bonnland | @dcherian | @jbusecke | @jhamman | @kmpaul

5.6.10 Intake-esm v2019.12.13

Features

- Add optional `preprocess` argument to *`to_dataset_dict()`* (GH#155) @matt-long
- Allow users to disable dataset aggregations by passing `aggregate=False` to *`to_dataset_dict()`* (GH#164) @matt-long
- Avoid manipulating dataset coordinates by using `data_vars=varname` when concatenating datasets via xarray {py:func}:`~xarray.concat()` (GH#174) @andersy005
- Support loading netCDF assets from openDAP endpoints (GH#176) @andersy005
- Add *`serialize()`* method to serialize collection/catalog (GH#179) @andersy005
- Allow passing extra storage options to the backend file system via *`to_dataset_dict()`* (GH#180) @bonnland
- Provide informational messages to the user via Logging module (GH#186) @andersy005

Bug Fixes

- Remove the caching option (GH#158) @matt-long
- Preserve encoding when aggregating datasets (GH#161) @matt-long
- Sort aggregations to make sure {py:func}:`~intake_esm.merge_util.join_existing` is always done before {py:func}:`~intake_esm.merge_util.join_new` (GH#171) @andersy005

Documentation

- Add example for preprocessing function (GH#168) @jbusecke
- Add FAQ style document to documentation (GH#182) & (GH#177) @andersy005 & @jhamman

Internal Changes

- Simplify group loading by using `concurrent.futures` (GH#185) @andersy005

Contributors to this release

(GitHub contributors page for this release)

@andersy005 | @bonnland | @dcherian | @jbusecke | @jhamman | @matt-long | @naomi-henderson | @Recalculate | @sebasblancogonz

5.6.11 Intake-esm v2019.10.15

Features

- Rewrite `intake-esm`'s core based on `(esm-collection-spec)_` Earth System Model Collection specification (GH#135) @andersy005, @matt-long, @rabernat

Breaking changes

- Replaced {py:class}:~intake_esm.core.esm_metadatastore with {py:class}:~intake_esm.core.esm_datastore, see the API reference for more details.
- `intake-esm` won't build collection catalogs anymore. `intake-esm` now expects an ESM collection JSON file as input. This JSON should conform to the Earth System Model Collection specification.

Contributors to this release

(GitHub contributors page for this release)

@aaronspring | @andersy005 | @bonnland | @dcherian | @n-henderson | @naomi-henderson | @rabernat

5.6.12 Intake-esm v2019.8.23

Features

- Add `mistral` data holdings to `intake-esm-datastore` (GH#133) @aaronspring
- Add support for `NA-CORDEX` data holdings. (GH#115) @jukent
- Replace `.csv` with `netCDF` as serialization format when saving the built collection to disk. With `netCDF`, we can record very useful information into the global attributes of the netCDF dataset. (GH#119) @andersy005
- Add string representation of `ESMMetadataStoreCatalog`` object ({pr}122`) @andersy005
- Automatically build missing collections by calling `esm_metadatastore(collection_name="GLADE-CMIP5")`. When the specified collection is part of the curated collections in `intake-esm-datastore`. (GH#124) @andersy005

```
In [1]: import intake

In [2]: col = intake.open_esm_metadatastore(collection_name="GLADE-CMIP5")

In [3]: # if "GLADE-CMIP5" collection isn't built already, the above is␣
↪equivalent to:

In [4]: col = intake.open_esm_metadatastore(collection_input_definition="GLADE-
↪CMIP5")
```

- Revert back to using official DRS attributes when building CMIP5 and CMIP6 collections. (GH#126) @andersy005
- Add `.df` property for interfacing with the built collection via dataframe To maintain backwards compatiblity. (GH#127) @andersy005

- Add `unique()` and `nunique()` methods for summarizing count and unique values in a collection. (GH#128) @andersy005

```
In [1]: import intake

In [2]: col = intake.open_esm_metadatastore(collection_name="GLADE-CMIP5")

In [3]: col
Out[3]: GLADE-CMIP5 collection catalogue with 615853 entries: > 3 resource(s)

          > 1 resource_type(s)

          > 1 direct_access(s)

          > 1 activity(s)

          > 218 ensemble_member(s)

          > 51 experiment(s)

          > 312093 file_basename(s)

          > 615853 file_fullpath(s)

          > 6 frequency(s)

          > 25 institute(s)

          > 15 mip_table(s)

          > 53 model(s)

          > 7 modeling_realm(s)

          > 3 product(s)

          > 9121 temporal_subset(s)

          > 454 variable(s)

          > 489 version(s)

In[4]: col.nunique()

resource 3
resource_type 1
direct_access 1
activity 1
ensemble_member 218
experiment 51
file_basename 312093
file_fullpath 615853
frequency 6
institute 25
mip_table 15
model 53
modeling_realm 7
```

(continues on next page)

(continued from previous page)

```
product 3
temporal_subset 9121
variable 454
version 489
dtype: int64

In[4]: col.unique(columns=['frequency', 'modeling_realm'])

{'frequency': {'count': 6, 'values': ['mon', 'day', '6hr', 'yr', '3hr', 'fx']},
'modeling_realm': {'count': 7, 'values': ['atmos', 'land', 'ocean', 'seaIce',
↪'ocnBgchem',
'landIce', 'aerosol']}}
```

Bug Fixes

- For CMIP6, extract `grid_label` from directory path instead of file name. (GH#127) @andersy005

Contributors to this release

(GitHub contributors page for this release)

5.6.13 Intake-esm v2019.8.5

Features

- Support building collections using inputs from intake-esm-datastore repository. (GH#79) @andersy005
- Ensure that requested files are available locally before loading data into xarray datasets. (GH#82) @andersy005 and @matt-long
- Split collection definitions out of config. (GH#83) @matt-long
- Add `intake-esm-builder`, a CLI tool for building collection from the command line. (GH#89) @andersy005
- Add support for CESM-LENS data holdings residing in AWS S3. (GH#98) @andersy005
- Sort collection upon creation according to order-by-columns, pass urlpath through stack for use in parsing collection filenames (GH#100) @pbranson

Bug Fixes

- Fix bug in `_list_files_hsi()` to return list instead of filter object. (GH#81) @matt-long and @andersy005
- `cesm._get_file_attrs` fixed to break loop when longest `stream` is matched. (GH#80) @matt-long
- Restore `non_dim_coords` to data variables all the time. (GH#90) @andersy005
- Fix bug in `intake_esm/cesm.py` that caused `intake-esm` to exclude hourly (1hr, 6hr, etc..) CESM-LE data. (GH#110) @andersy005
- Fix bugs in `intake_esm/cmip.py` that caused improper regular expression matching for `table_id` and `grid_label`. (GH#113) & (GH#111) @naomi-henderson and @andersy005

Internal Changes

- Refactor existing functionality to make intake-esm robust and extensible. (GH#77) @andersy005
- Add `aggregate._override_coords` function to override dim coordinates except time in case there's floating point precision difference. (GH#108) @andersy005
- Fix CESM-LE ice component peculiarities that caused intake-esm to load data improperly. The fix separates variables for `ice` component into two separate components:
  - `ice_sh`: for southern hemisphere
  - `ice_nh`: for northern hemisphere

  (GH#114) @andersy005

Contributors to this release

(GitHub contributors page for this release)

5.6.14 Intake-esm v2019.5.11

Features

- Add implementation for The Gridded Meteorological Ensemble Tool (GMET) data holdings (GH#61) @andersy005
- Allow users to specify exclude*dirs for CMIP collections (GH#63) & (GH#62) @andersy005
- Keep CMIP6 `tracking_id` in `merge_keys` (GH#67) @andersy005
- Add implementation for ERA5 datasets (GH#68) @andersy005

Contributors to this release

(GitHub contributors page for this release)

5.6.15 Intake-esm v2019.4.26

Features

- Add implementations for `CMIPCollection` and `CMIPSource` (GH#38) @andersy005
- Add support for CMIP6 data (GH#46) @andersy005
- Add implementation for The Max Planck Institute Grand Ensemble (MPI-GE) data holdings (GH#52) & (GH#51) @aaronspring and @andersy005
- Return dictionary of datasets all the time for consistency (GH#56) @andersy005

Bug Fixes

- Include multiple netcdf files in same subdirectory (GH#55) & (GH#54) @naomi-henderson and @andersy005

Contributors to this release

(GitHub contributors page for this release)

5.6.16 Intake-esm v2019.2.28

Features

- Allow CMIP integration (GH#35) @andersy005

Bug Fixes

- Fix bug on build catalog and move `exclude_dirs` to `locations` (GH#33) @matt-long

Internal Changes

- Change Logger, update dev-environment dependencies, and formatting fix in input.yml (GH#31) @matt-long
- Update CircleCI workflow (GH#32) @andersy005
- Rename package from `intake-cesm` to `intake-esm` (GH#34) @andersy005

PYTHON MODULE INDEX

INDEX