catalog$_{m}aker Documentation$

Release 0.1.0

Eleanor Smith

Dec 13, 2021

CONTENTS:

CHAPTER

ONE

CATALOG MAKER

A package to build intake catalogs for cmip5, cmip6 and cordex data holdings

- Free software: BSD - see LICENSE file in top-level package directory

1.1 Installing

Create a clone of the repository:

```
git clone https://github.com/roocs/catalog-maker.git
```

go into the catalog-maker directory

```
cd catalog-maker
```

Install the package into your virtual environment:

```
pip install -e .
```

1.2 Creating an Intake Catalog

Catalog maker provides tools for writing data catalogs of the known data holdings in a csv format, described by a YAML file.

For each project in `catalog_maker/etc/roocs.ini` there are options to set the file paths for the inputs and outputs of this catalog maker. A list of datasets to include needs to be provided. The path to this list for each project can be set in `catalog_maker/etc/roocs.ini`. The datasets in this list must be what you want in the `ds_id` column of the csv file.

If creating a c3s-cmip6 inventory, make sure the dataset ids start with 'c3s-cmip6' instead of CMIP6

The data catalog is created using a database backend to store the results of the scans, from which the csv and YAML files will be created. For this, a postgresql database is required. Once you have a database, you need to export an environment variable called `$ABCUNIT_DB_SETTINGS`:

```
$ export ABCUNIT_DB_SETTINGS="dbname=<name> user=<user> host=<host> password=<pwd>"
```

The table created will be named after the project you are creating a catalog for in the format `<project_name>_catalog_results` e.g. c3s_cmip6_catalog_results

For the below commands you must be in the catalog-maker directory

1.3 Creating batches

Once the list of datasets is collated a number of batches must be created:

```
$ python catalog_maker/cli.py create-batches -p c3s-cmip6
```

The option `-p` is required to specify the project.

1.4 Creating catalog entries

Once the batches are created, the catalog maker can be run - either locally or on lotus. The settings for how many datasets to be included in a batch and the maximum duration of each job on lotus can also be changed in `catalog_maker/etc/roocs.ini`.

Each batch can be run idependently, e.g. running batch 1 locally:

```
$ python catalog_maker/cli.py run -p c3s-cmip6 -b 1 -r local
```

or running all batches on lotus:

```
$ python catalog_maker/cli.py run -p c3s-cmip6 -r lotus
```

This creates a table in the database containing an ordered dictionary of the entry for each file in each dataset if successful, or the error traceback if there is an Exception raised.

It is also possible to force a rescan of datasets that have already been scanned. To do this use the *-f* flag.

e.g.

```
$ python catalog_maker/cli.py run -p c3s-cmip6 -r lotus -f
```

1.5 Viewing entries and errors

To view the records:

```
$ python catalog_maker/cli.py list -p c3s-cmip6
```

With many entries, this may take a while.

To just get a count of how many files have been scanned:

```
$ python catalog_maker/cli.py list -p c3s-cmip6 -c
```

To see any errors:

```
$ python catalog_maker/cli.py show-errors -p c3s-cmip6
```

To see just a count of errors:

```
$ python catalog_maker/cli.py show-errors -p c3s-cmip6 -c
```

Each count will show how many files and how many datasets have been successful/failed.

The list count will also show the total numbers of datasets/files in the database - including errors. The error count will show whether there are any datasets that have files which have succeeded and failed i.e. that are partially scanned. You can then use the delete command explained below to delete the entries for these partially scanned datasets if required.

1.6 Deleting entries

It is possible to delete entries by dataset id:

```
$ python catalog_maker/cli.py delete -p c3s-cmip6 -d <ds_id>
```

You can also provide a list of dataset ids to the -d option. This command only deletes successful entries and will leave errors for the datasets specified in the database.

To delete all entries, including errors for specific dataset ids, use the command:

```
$ python catalog_maker/cli.py delete -p c3s-cmip6 -d <ds_id> -e
```

1.7 Writing to CSV

The final command is to write the entries to a csv file.

```
$ python catalog_maker/cli.py write -p c3s-cmip6
```

The flag -c will compress the output csv file. e.g.

```
$ python catalog_maker/cli.py write -p c3s-cmip6 -c
```

The csv file will be generated in the `csv_dir` specified in `catalog_maker/etc/roocs.ini` and will have the name "{project}_{version_stamp}.csv.gz" if compressed e.g. `c3s-cmip6_v20210414.csv.gz` or "{project}_{version_stamp}.csv" if not compressed.

A yaml file will be created the `catalog_dir` specified in `catalog_maker/etc/roocs.ini`. It will have the name `c3s.yml` and will contain the below for each project scanned and which is using the same `catalog_dir`:

```
sources:
  c3s-cmip6:
    args:
      urlpath:
    csv_kwargs:
      blocksize: null
      compression: gzip
      dtype:
        level: object
    description: c3s-cmip6 datasets
    driver: intake.source.csv.CSVSource
    metadata:
      last_updated:
```

`urlpath` and `last_updated` for a project will be updated very time the csv file is written for the project.

1.8 Deleting the table of results

In order to delete all entries in the table of results:

```
$ python catalog_maker/cli.py clean -p c3s-cmip6
```

1.9 Credits

This package was created with `Cookiecutter` and the `audreyr/cookiecutter-pypackage` project template.

- Cookiecutter: https://github.com/audreyr/cookiecutter
- cookiecutter-pypackage: https://github.com/audreyr/cookiecutter-pypackage

CHAPTER

TWO

INSTALLATION

2.1 From sources

The sources for catalog_maker can be downloaded from the Github repo.

Clone the public repository:

```
$ git clone git://github.com/roocs/catalog_maker
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER

THREE

USAGE

To use catalog_maker in a project:

```
import catalog_maker
```

For information on the configuration options available in roocs-utils, see: https://roocs-utils.readthedocs.io/en/latest/configuration.html#catalog-maker

CHAPTER

FOUR

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/roocs/catalog_maker/issues.

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.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

catalog_maker could always use more documentation, whether as part of the official catalog_maker docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/roocs/catalog_maker/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.

4.2 Get Started!

Ready to contribute? Here's how to set up *catalog_maker* for local development.

1. Fork the *catalog-maker* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/catalog-maker.git
```

3. Install your local copy into a virtualenv.

```
$ python venv venv
$ source venv/bin/activate
$ cd catalog_maker/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 catalog_maker tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.6, 3.7 and 3.8. Check https://travis-ci.com/roocs/catalog_maker/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ pytest tests.test_catalog_maker
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch (or major/minor)
$ git push
$ git push --tags
```

Then to upload to pypi:

```
$ python setup.py sdist bdist_wheel
$ twine upload dist/*
```

CHAPTER

FIVE

CREDITS

5.1 Development Lead

- Eleanor Smith <eleanor.smith@stfc.ac.uk>

5.2 Contributors

None yet. Why not be the first?

CHAPTER

SIX

HISTORY

6.1 0.1.0 (unreleased)

- First release.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search