Onyx | JupyterLab Extension

CLIMB-TRE

None

Table of contents

1. JupyterLab Extension

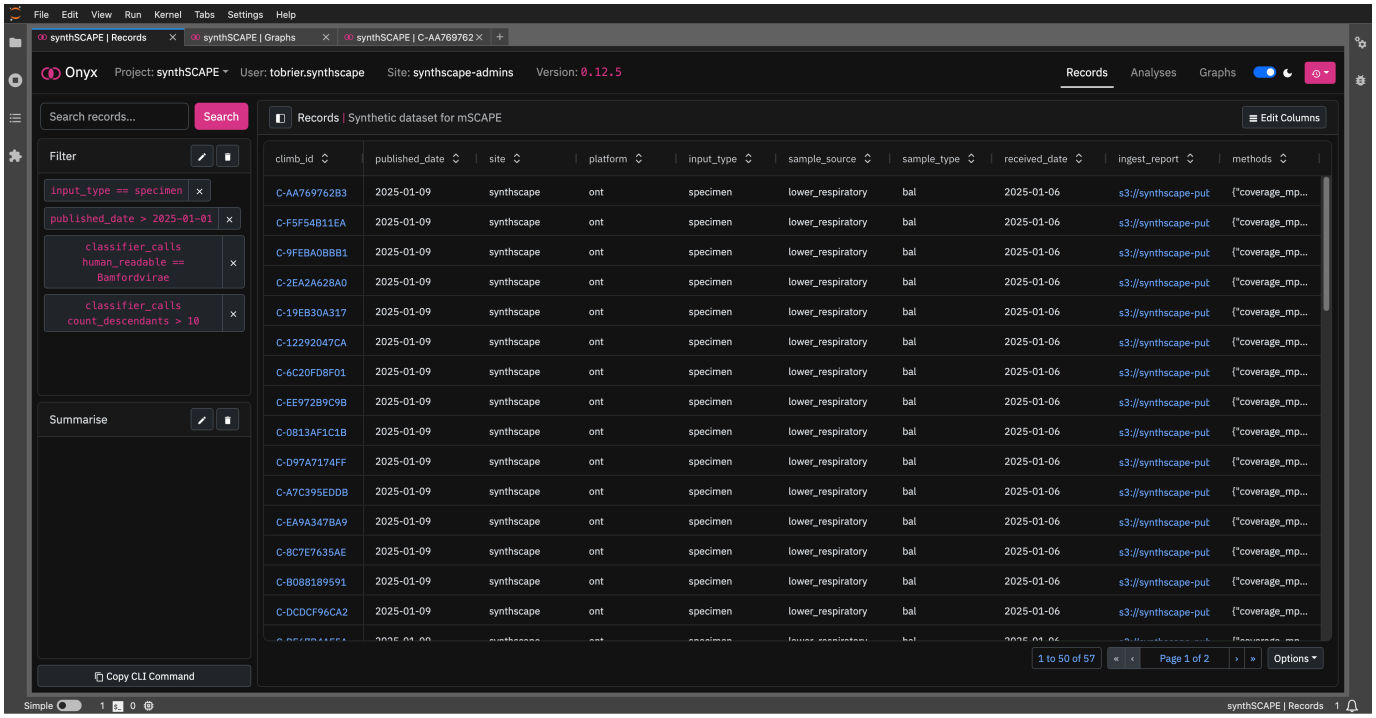
1.1 JupyterLab Extension for Onyx

1.1.1 Introduction

This is the documentation for Onyx-extension, an extension to JupyterLab that provides:

- : Access to the Onyx GUI, a graphical user interface for interacting with the Onyx database.
- : Links to the CLIMB-TRE documentation.

Onyx is being developed as part of the CLIMB-TRE project.



1.1.2 Contents

1.2 Installation

Guidance for installing the Onyx JupyterLab extension, or building it manually for development.

Usage within CLIMB JupyterLab Servers

If you are running a CLIMB JupyterLab server, you **do not** need to install the extension, as it comes pre-configured in your environment.

If you cannot see the most up-to-date version of the Onyx JupyterLab extension, this is because you will have previously installed your own version manually.

To revert your Onyx JupyterLab extension to the managed up-to-date version, navigate to your terminal and run:

```
$ pip uninstall climb-onyx-gui
```

And restart your JupyterLab server.

1.2.1 Install from PyPI

Assuming you have JupyterLab installed:

```
$ pip install climb-onyx-gui
```

Otherwise:

```
$ pip install "jupyterlab>=3" climb-onyx-gui
```

1.2.2 Build from source

Clone the repository:

```
$ git clone https://github.com/CLIMB-TRE/onyx-extension.git
$ cd onyx-extension/
```

Ensure you have Miniconda (or an alternative conda installer) available. Installation instructions for Conda can be found here.

Create and activate a conda environment with JupyterLab and NodeJS:

```
$ conda create -n jupyterlab-ext -c conda-forge jupyterlab=4 nodejs=20
$ conda activate jupyterlab-ext
```

Install the extension dependencies with the JupyterLab package manager `jlpm`:

```
$ jlpm install
```

Build the extension and install it:

```
$ jlpm run build
$ pip install -ve .
```

Optionally, copy and edit `.env.example` with `ONYX_DOMAIN` and `ONYX_TOKEN` for your development instance of Onyx:

```
$ cp .env.example .env
$ source .env # After editing
```

You can now launch JupyterLab with:

```
$ jupyter lab
```

And the Onyx extension will be ready on the launcher.

1.2.3 Local development

If you wish to develop the extension, ensure you have followed the above steps to build, install and run the extension from source.

From there, you can simply modify the extension code and dependencies, and reinstall/rebuild the extension by executing:

```
$ jlpm install && jlpm run build && pip install -ve .
```

and then relaunching JupyterLab.

1.2.4 Troubleshooting

If you are seeing the frontend extension, but it is not working, check that the server extension is enabled with:

```
jupyter server extension list
```

If the server extension is installed and enabled, but you are not seeing the frontend extension, check the frontend extension is installed with:

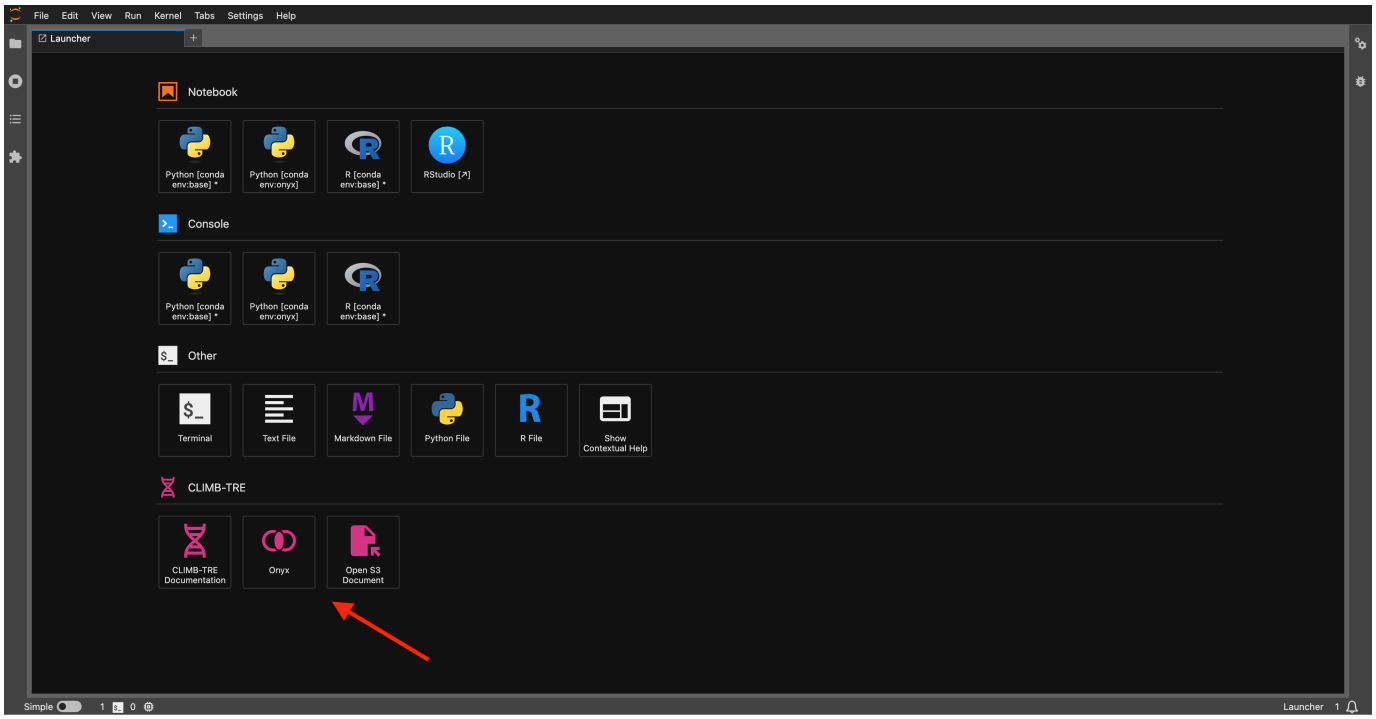
```
jupyter labextension list
```

1.3 Graphical User Interface

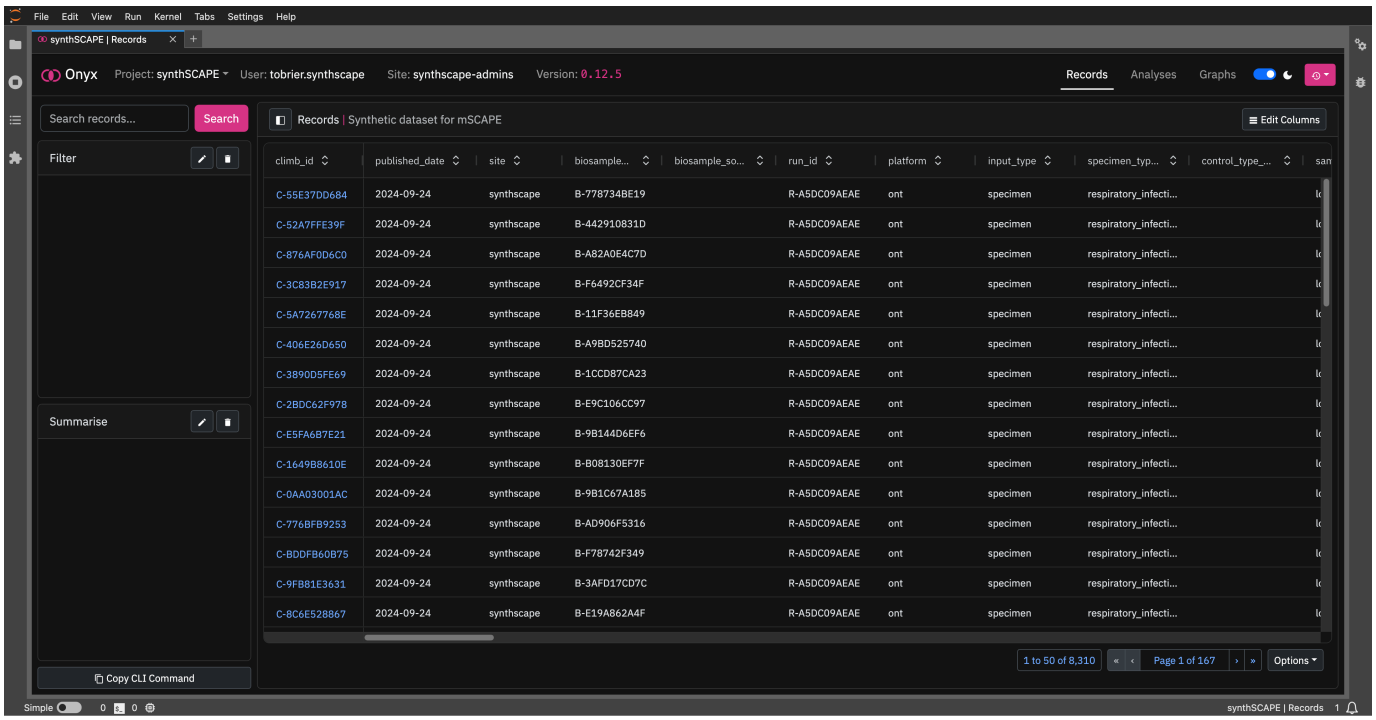
1.3.1 Quick Start

Launching Onyx

To open a new instance of Onyx, open the JupyterLab launcher and click the Onyx panel:



This will open Onyx:



> **Tip**
>
> You can open multiple instances of Onyx, and these will be persisted within your current JupyterLab workspace.

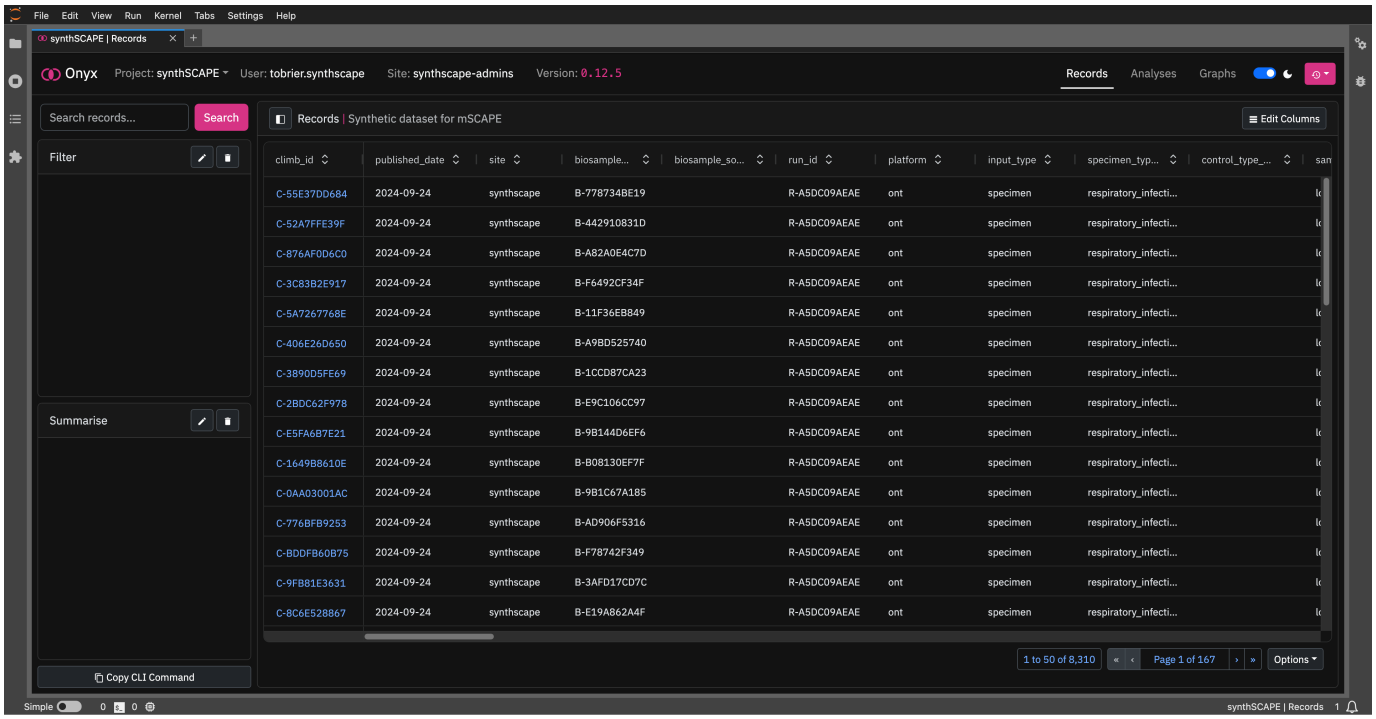
Navigating the Header

Looking at the header, from left to right we have the following options:

- `Project: synthSCAPE`
- This dropdown shows the current active project. Click this to select a different project.
- `User: tobrier.synthscape`
- This shows the user you are authenticated as. Click this to view the profile page.
- `Site: synthscape-admins`
- This shows the site you are a member of. Click this to view the site page.
- `Version: 0.12.5`
- This shows the version of the JupyterLab extension that is currently running.
- `Records`
- Click this to view records for the current project.
- `Analyses`
- Click this to view analyses for the current project.
- `Graphs`
- Click this to view the graph dashboard for the current project.
-
- Click this slider to switch between light and dark mode for Onyx.
-
- Click this to view your most recently viewed records and analyses.

1.3.2 Querying Data

Records and analyses share the same interface and functionality for browsing.



This includes:

- A `Search` bar, for basic substring matching against visible fields.
- The `Filter` panel, for adding filters against visible, non-visible and nested fields.
- The `Summarise` panel, for computing the number of items per summary group, as specified by the selected fields.
- The `Copy CLI Command`, for copying the current filters and summary fields into a command for the Onyx command-line interface.
- The results panel, `Records` or `Analyses`, that displays results matching the current query. Here, results can be re-ordered, as well as exported via the table `Options` dropdown. The `Edit Columns` button can also be used to change the returned columns.

We will now learn how to filter, aggregate and export data using this interface.

Filtering Data

Info

`synthSCAPE` is an Onyx project within CLIMB-TRE for hosting simulated metagenomic data as part of mSCAPE, a world-leading initiative trialling the use of metagenomic data for public health surveillance.

You can find the metadata schema for the synthSCAPE Onyx project here.

DEFINING THE QUERY

We are going to add filters on the `synthSCAPE` dataset to solve the following problem:

> **Query**
>
> Match all synthSCAPE records from 2025 that have a sequence purpose, a run ID of either `R-14EC71EBA7` or `R-F42A056185`, and contain at least 100 reads of `Influenza A Virus` (taxon ID `11320`).

This query can be broken down into the following criteria:

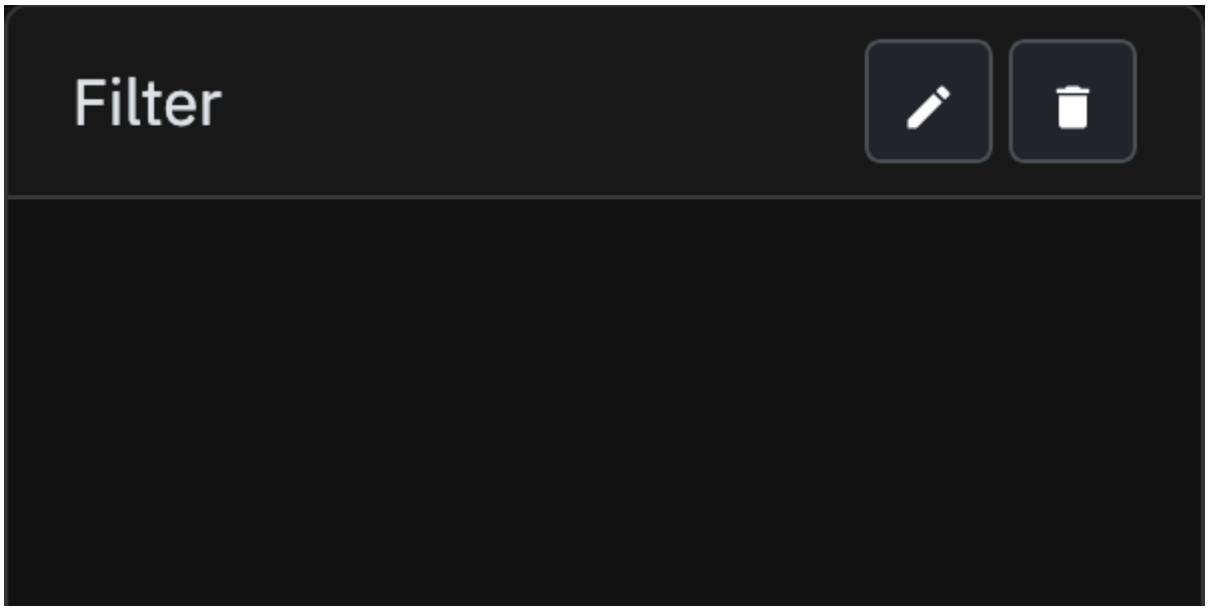
1. The `published_date` must be greater than or equal to `2025-01-01`.
2. The `sequence_purpose` must not be blank.
3. The `run_id` must be either `R-14EC71EBA7` or `R-F42A056185`.
4. Each record's `classifier_calls` must contain *at least* one entry matching the condition:

- `(taxon_id == 11320) AND (count_descendants >= 100)`

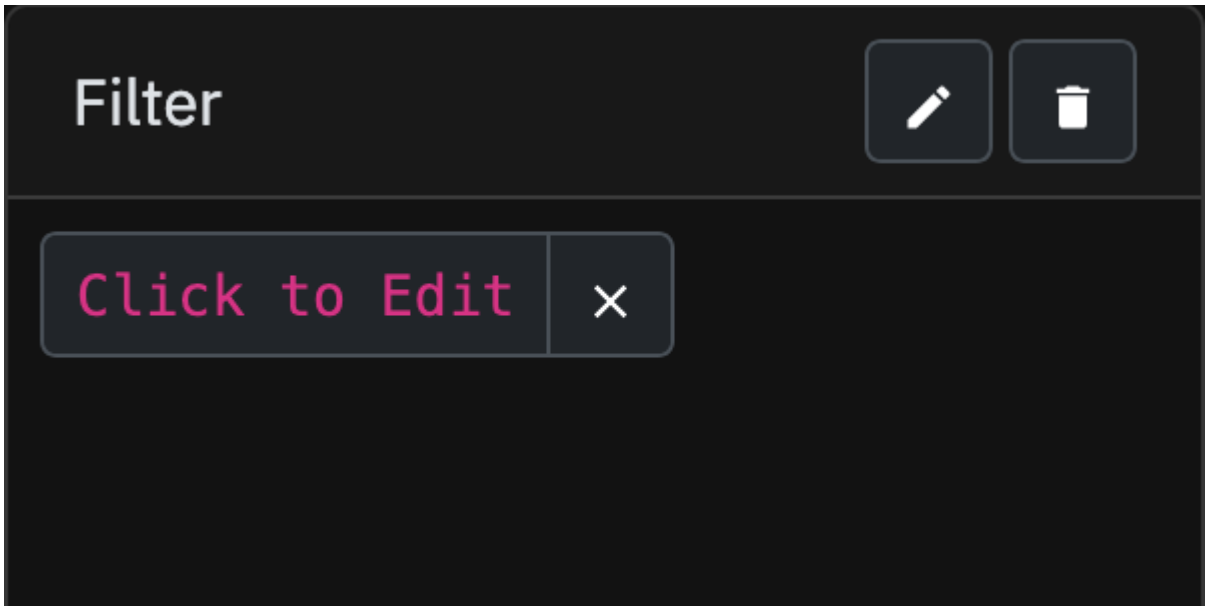
BUILDING THE QUERY

Adding a Filter

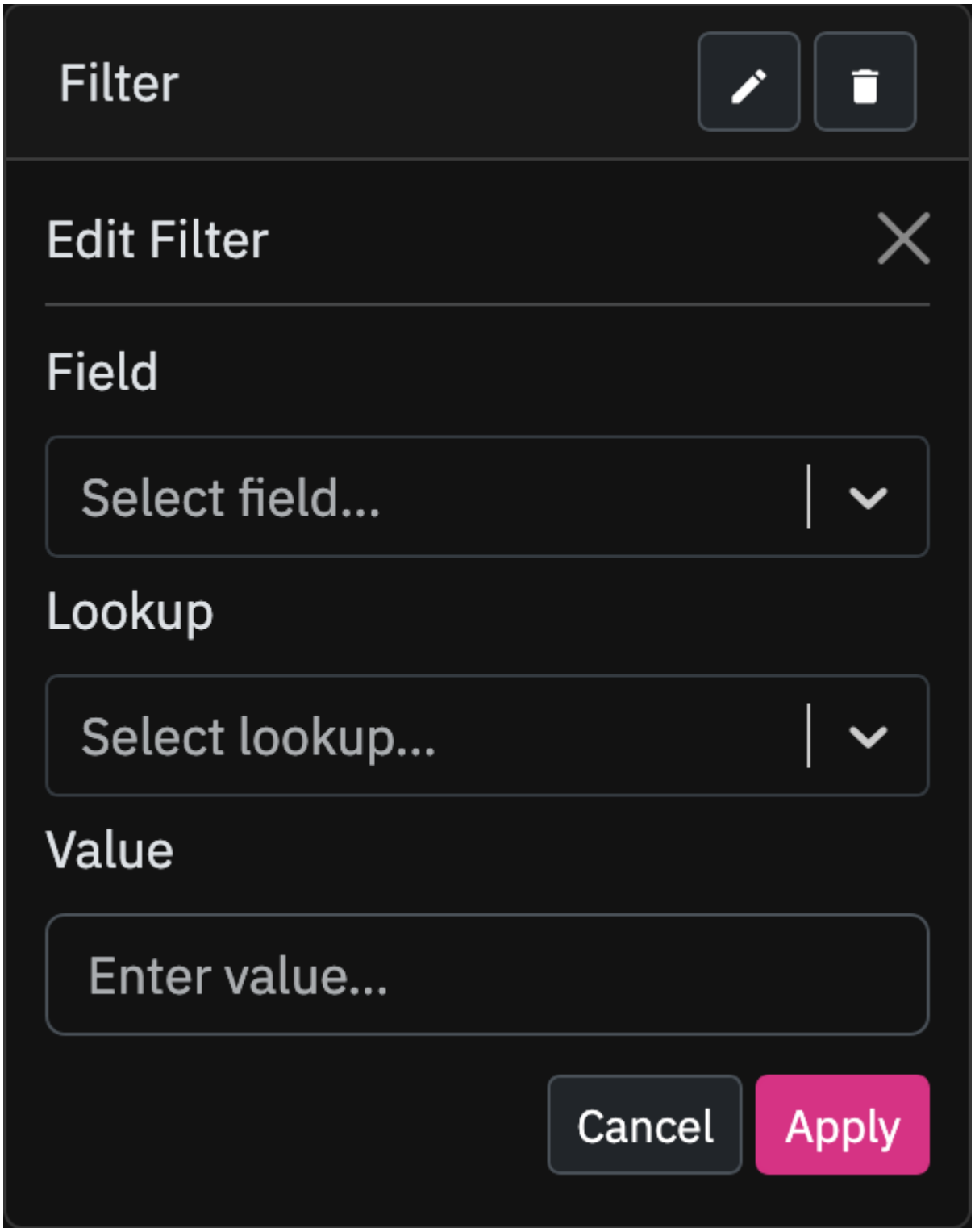
To add a new filter, click the ✎ icon on the `Filter` panel:



This creates an empty filter with the title `Click to Edit`.

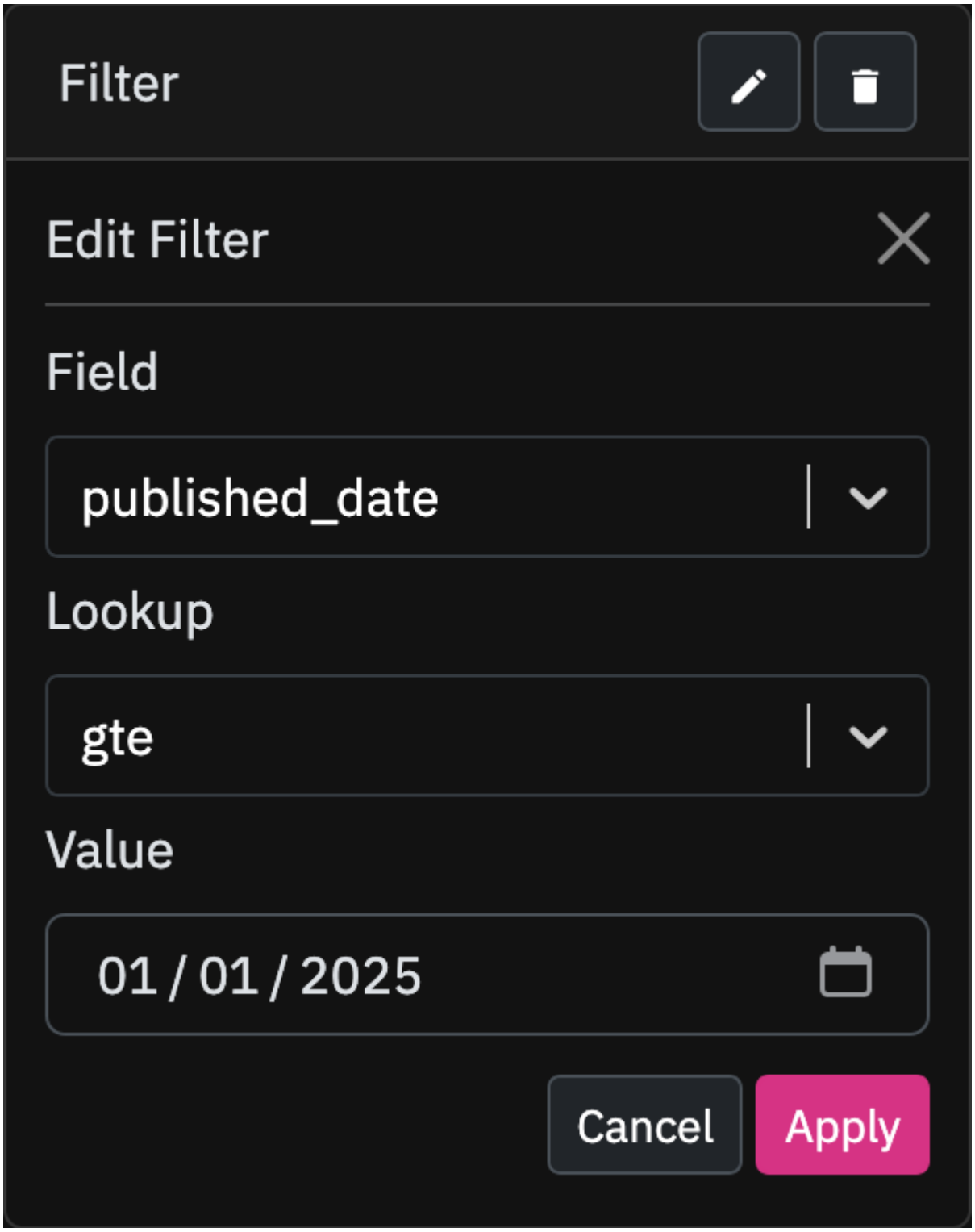


Clicking on this filter will open its settings:

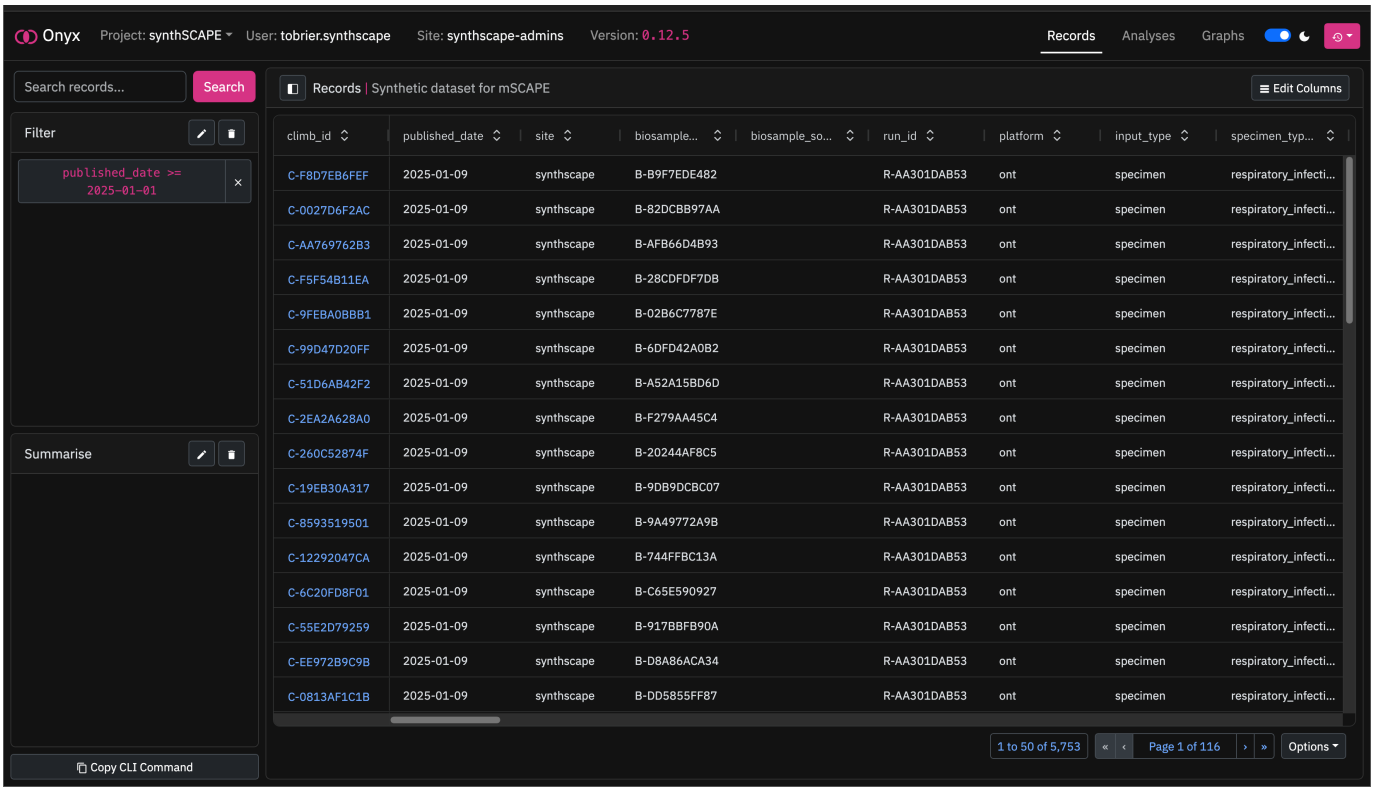


We will edit the filter as following:

- Set the field to `published_date`.
- Set the lookup to `gte` (greater than or equal).
- Set the value to `2025-01-01`.



Then hit `Apply` to filter to the dataset:



As we can see, the dataset has been filtered to return only records with a `published_date` greater than or equal to `2025-01-01`.

> **Tip**
>
> In this example, we have used the `gte` (greater than or equal) lookup for `published_date`. However, we could also use the `iso_year` lookup and set this to `2025` instead.

More Filters

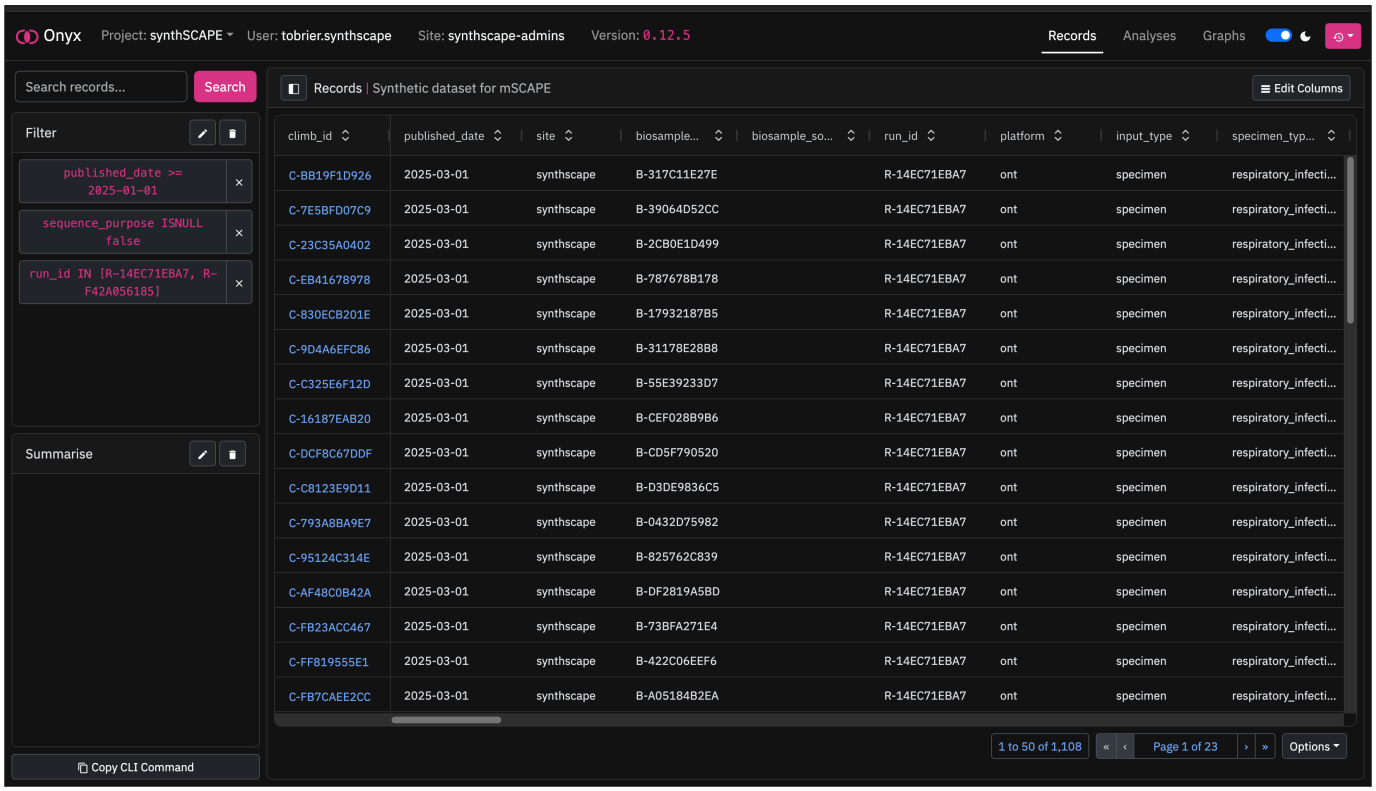
To add the second filter, we create another filter with:

- Field: `sequence_purpose`
- Lookup: `isnull`
- Value: `false`

And to add the third filter, we create another filter with:

- Field: `run_id`
- Lookup: `in`
- Values: `R-14EC71EBA7`, `R-F42A056185`

The dataset has now been filtered further:



Nested Filters

To match the final requirement, we need to create two **nested filters**:

Nested filter 1:

- Field: `classifier_calls__taxon_id`
- Lookup: `exact`
- Value: `11320`

Nested filter 2:

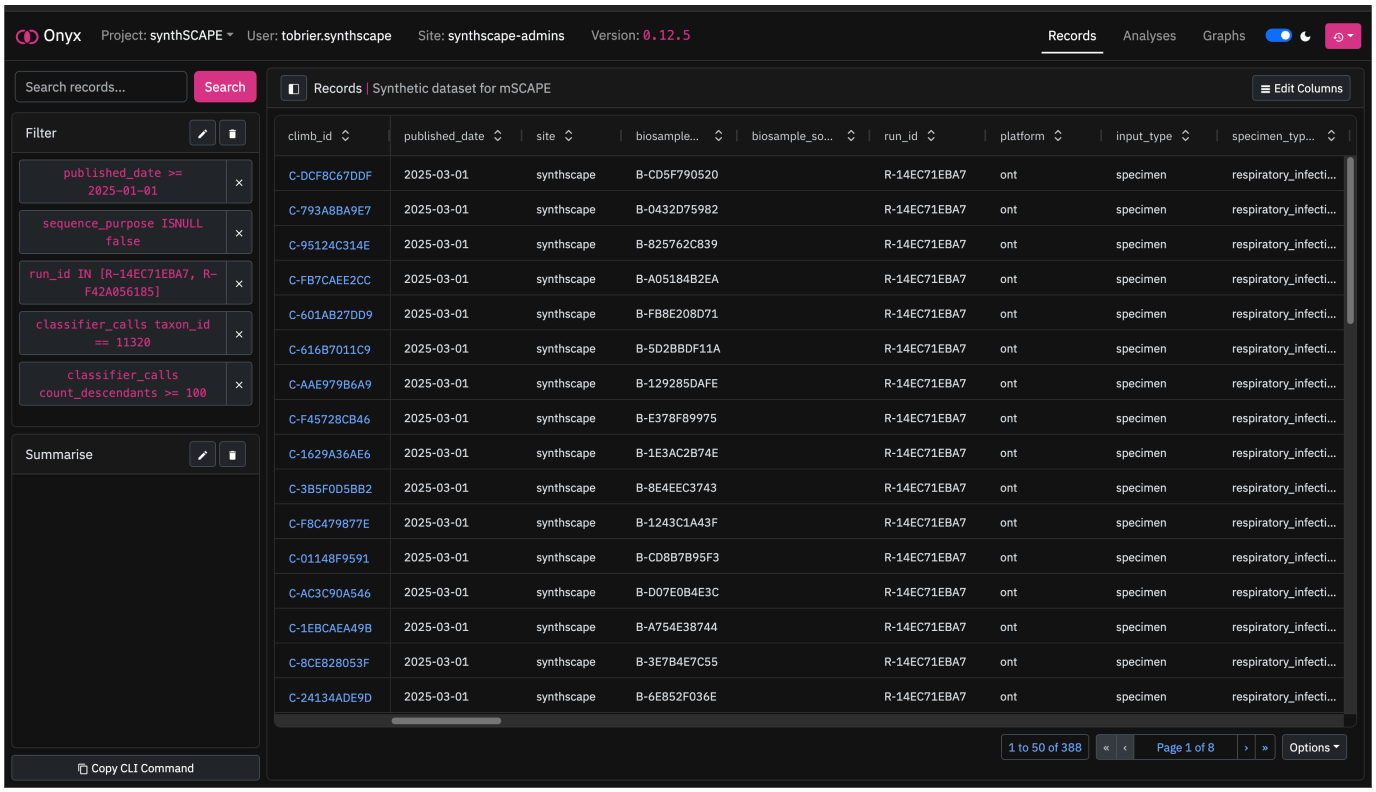
- Field: `classifier_calls__count_descendents`
- Lookup: `gte`
- Value: `100`

Each record in `synthSCAPE` contains multiple `classifier_calls` entries, that correspond to the taxa identified by Kraken2 within the sample. These `classifier_calls` entries also contain information such as the number of reads matched to each taxon, as well as the taxonomic rank.

When we apply the first nested filter to this `classifier_calls` table, Onyx will return all records which have *at least* one `classifier_call` with `taxon_id == 11320`.

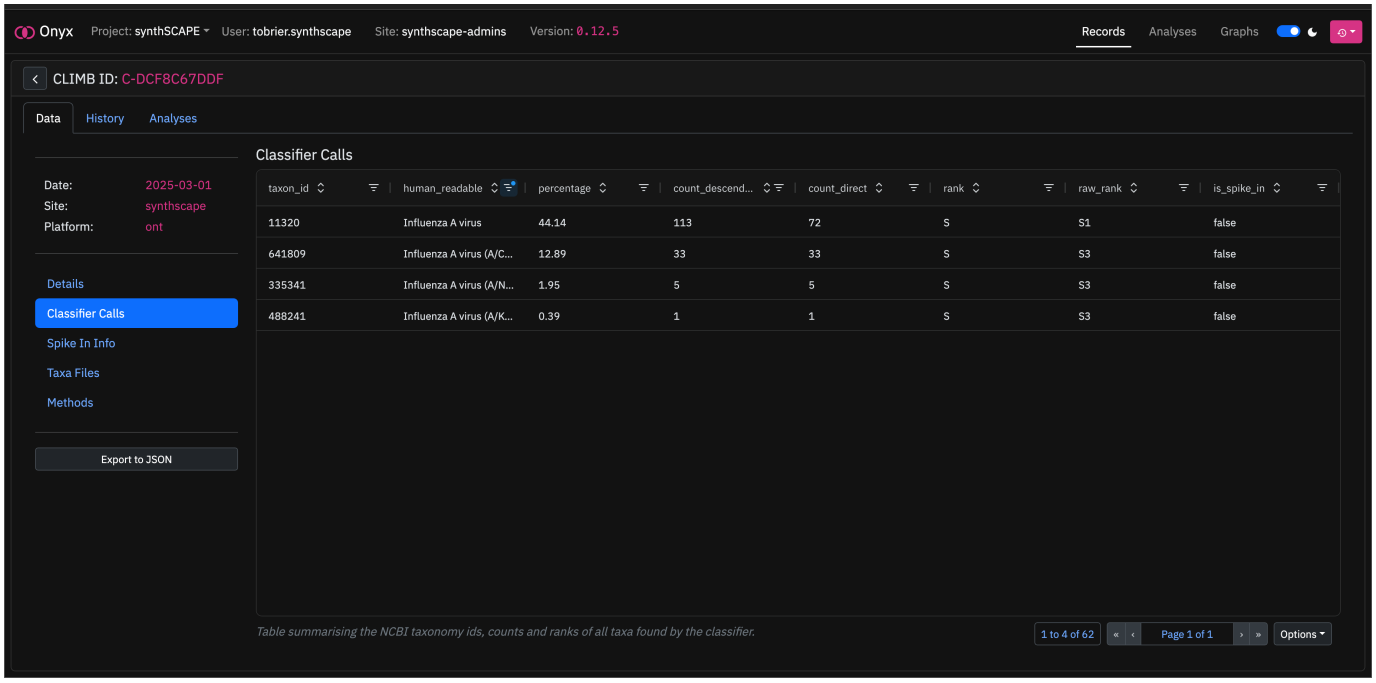
When we apply both nested filters, Onyx will return all records which have *at least* one `classifier_calls` entry matching **both** `taxon_id == 11320` and `count_descendents >= 100`.

QUERY RESULTS



As we can see, we have now filtered the dataset from the initial 8310 records down to just 388 records that match the target query.

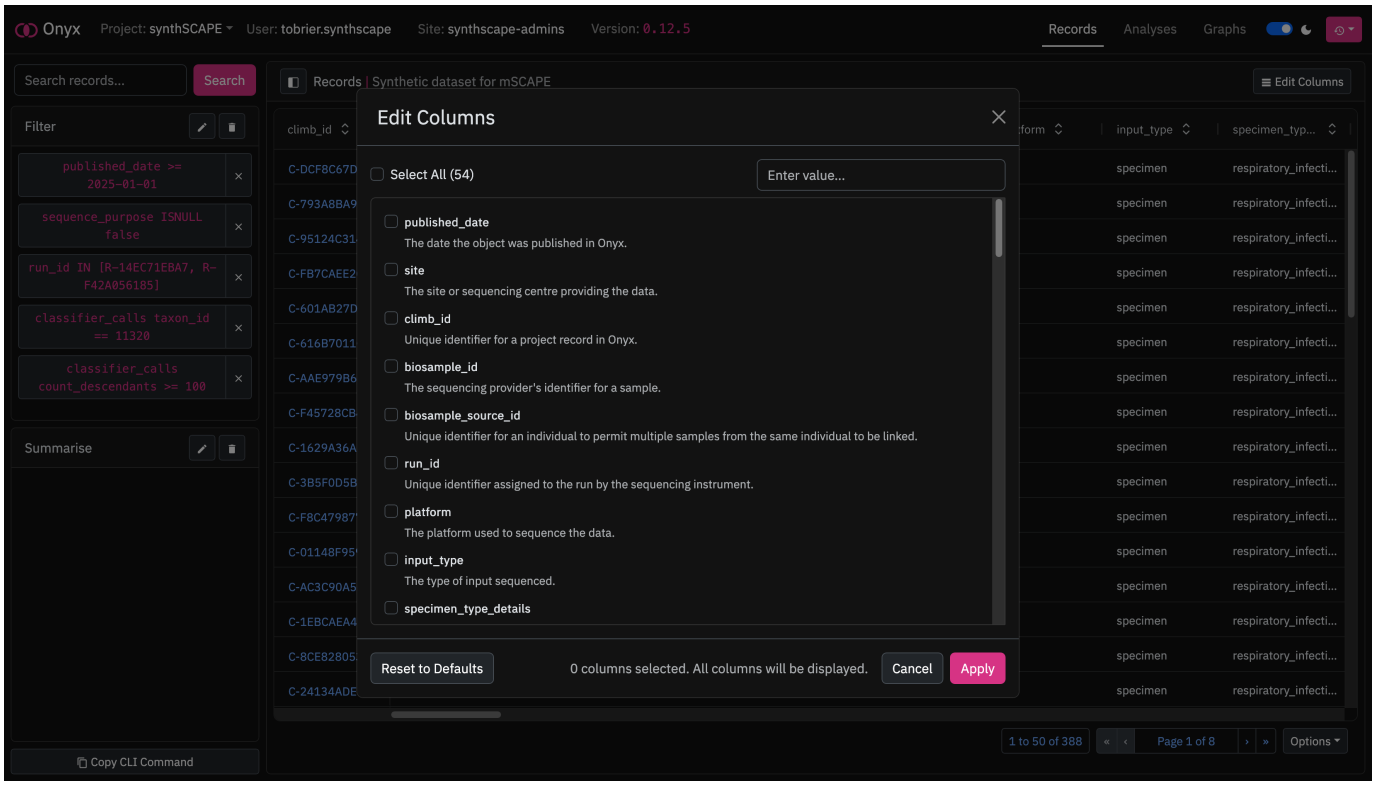
We can click on the first record, `C-DCF8C67DDF`, and navigate down to the `Classifier Calls` table and confirm over 100 reads matching `Influenza A Virus`:



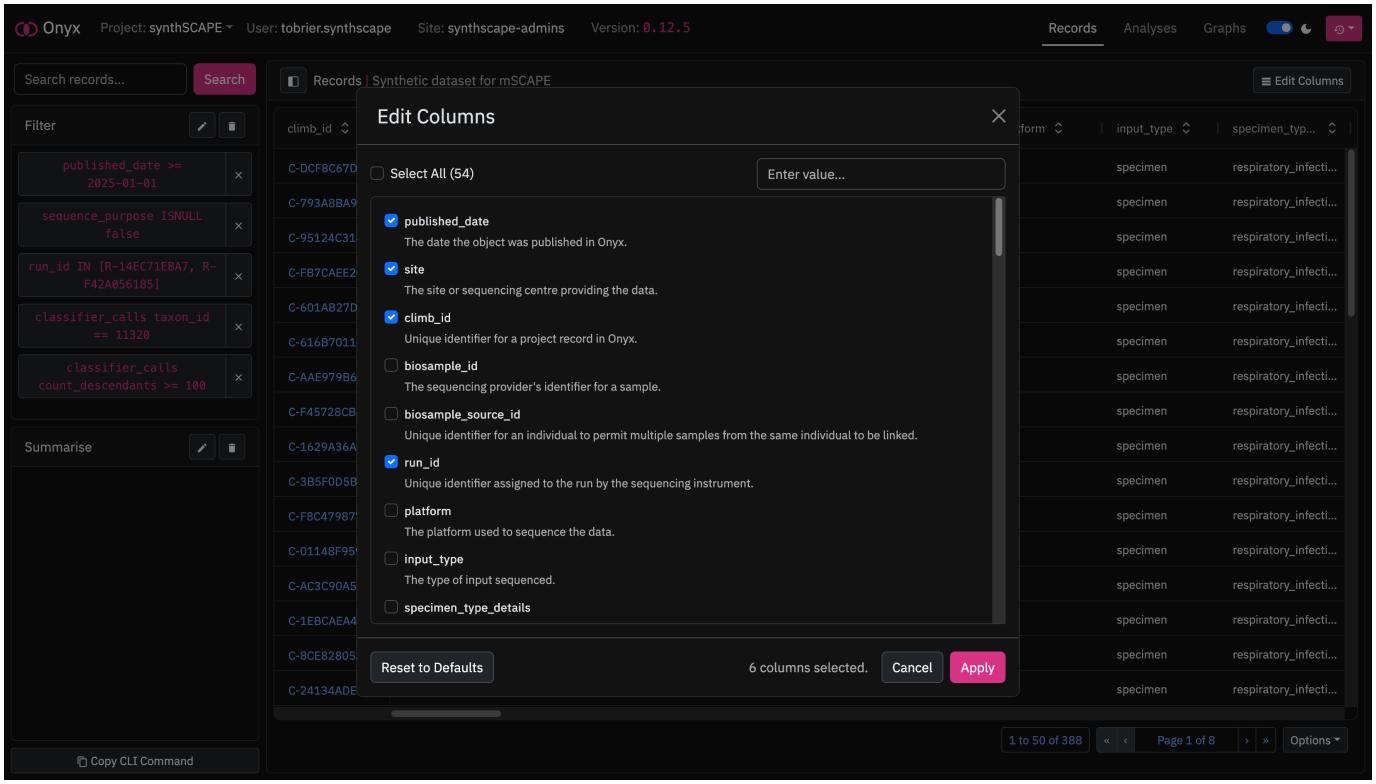
EDITING COLUMNS

Going back to the results view, we may want to fix the columns that get returned from filtering. This will also speed up any later exports, as we are only exporting the columns that we need.

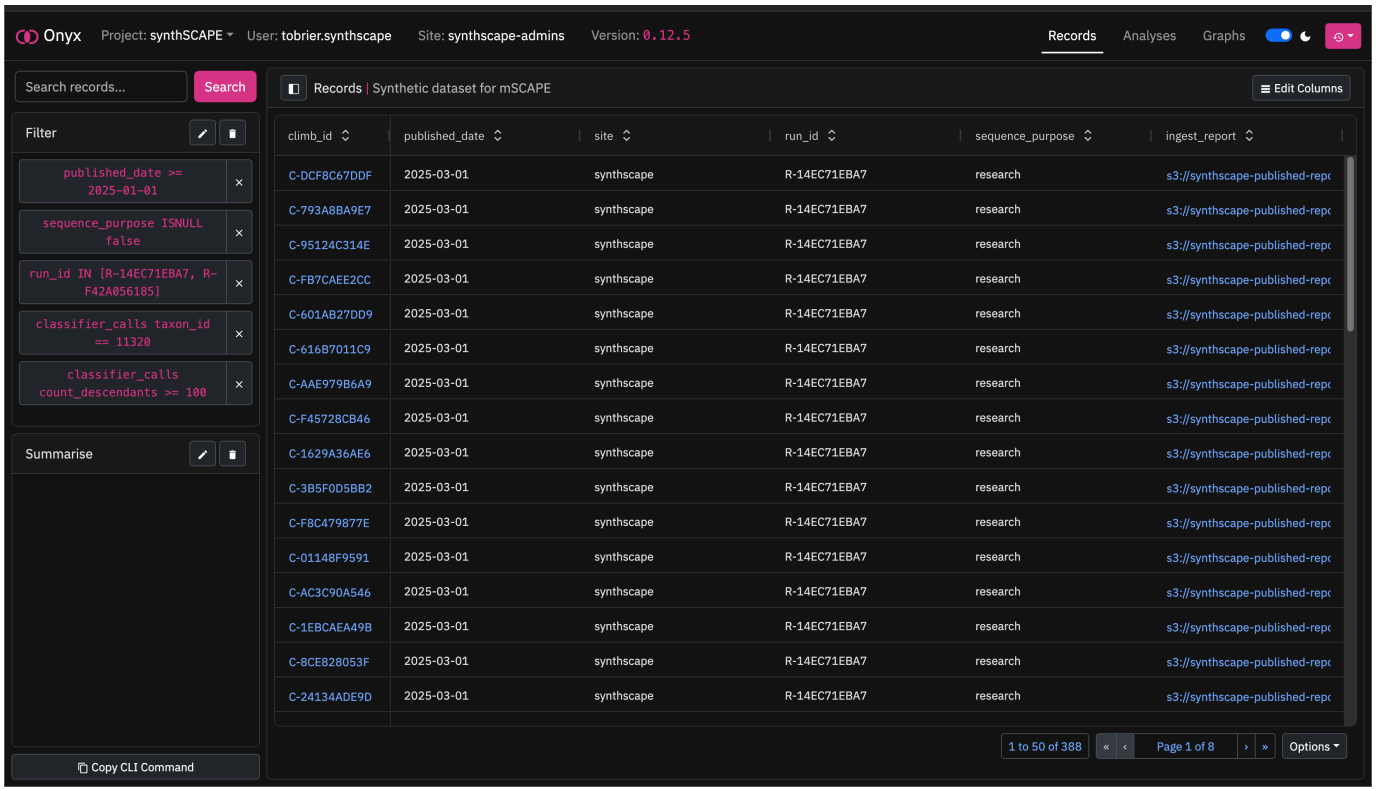
We can open the columns editor by clicking the `Edit Columns` button:



We will select the columns `climb_id`, `site`, `published_date`, `sequence_purpose` and `ingest_report`:



Then hit `Apply`, and only the columns we are interested in are returned:

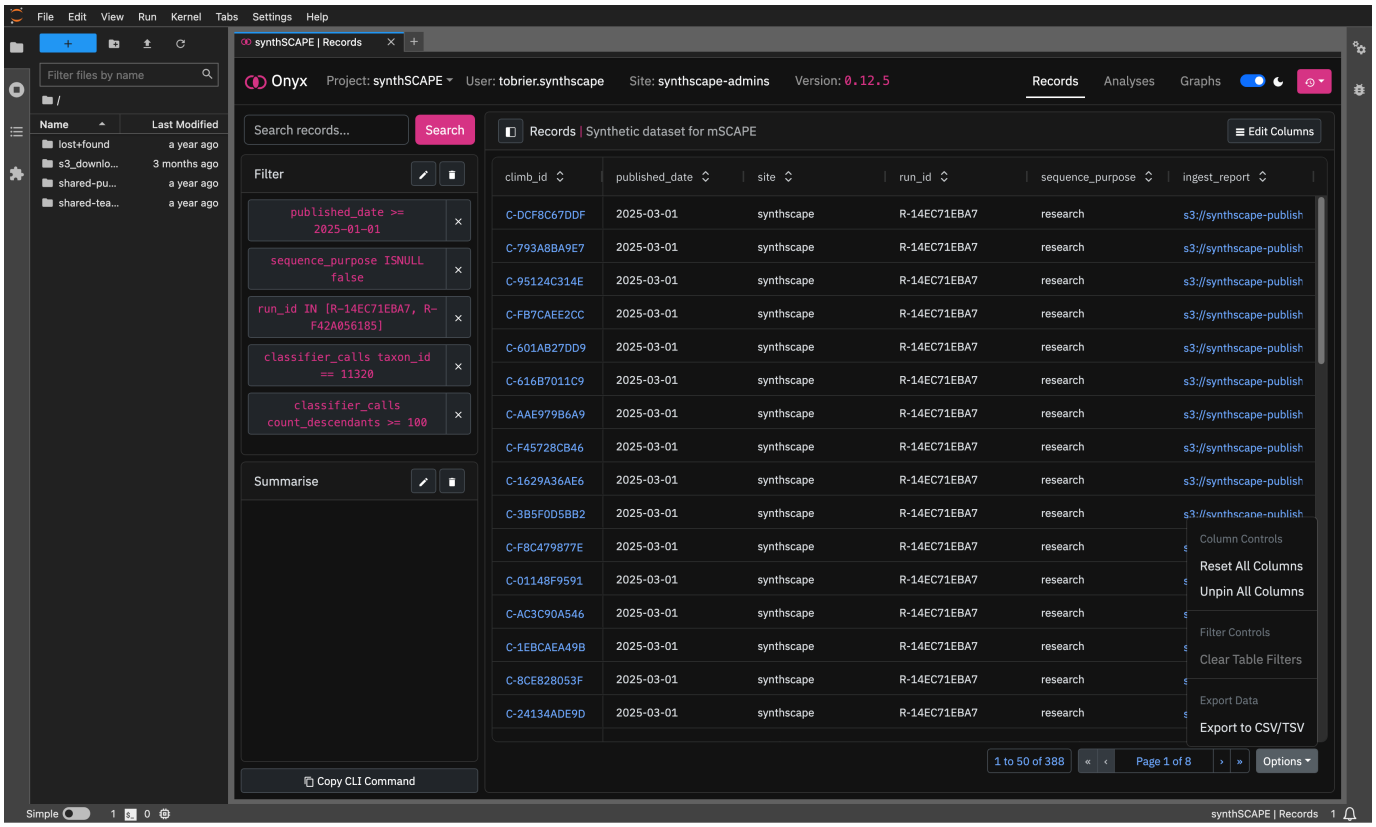


EXPORTING RESULTS

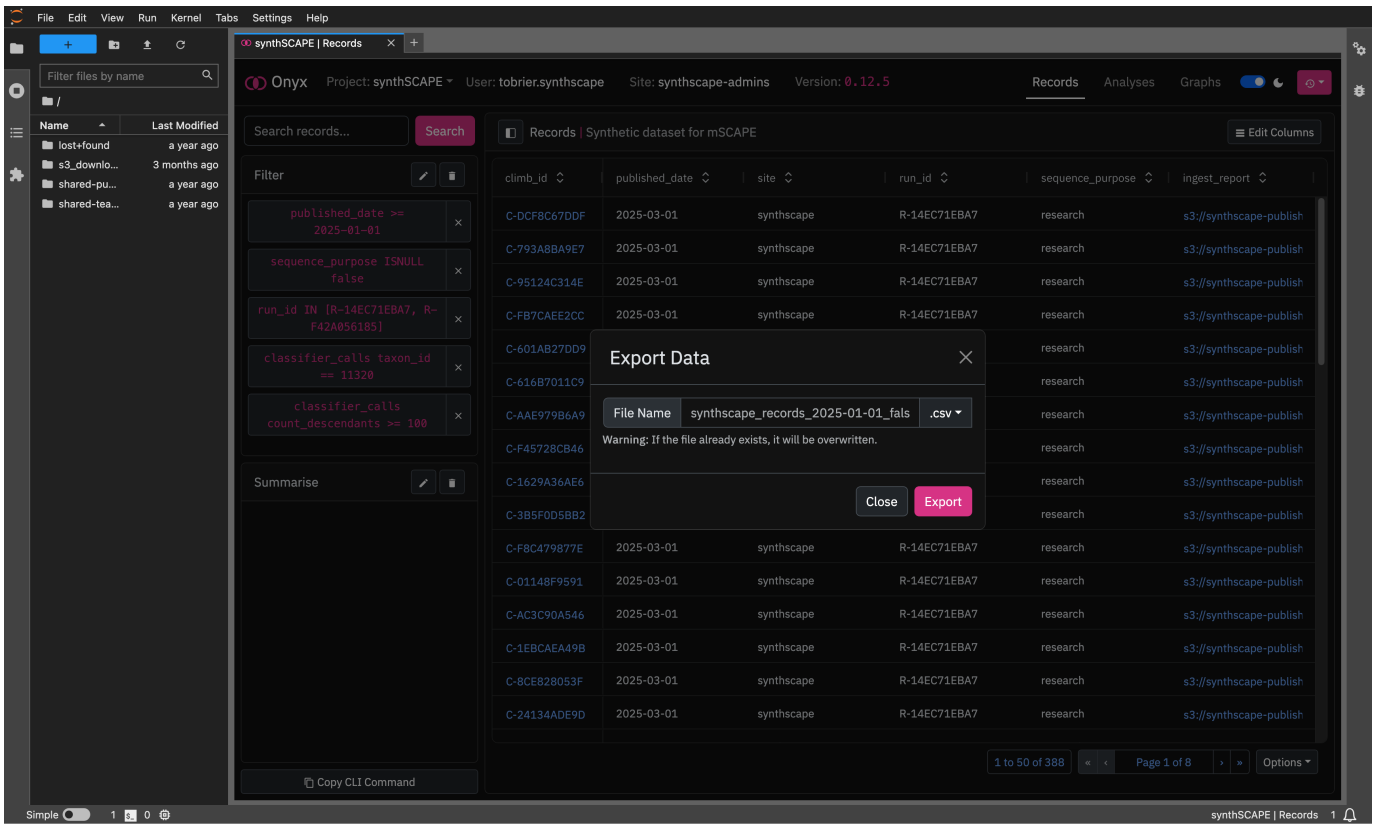
Exporting to CSV/TSV

We have our filtered records and the columns we are interested in. Now, we want to export this data to a CSV or TSV within JupyterLab for continued analysis.

Navigate down to the `Options` menu at the bottom-right of the results table, and select `Export to CSV/TSV`:

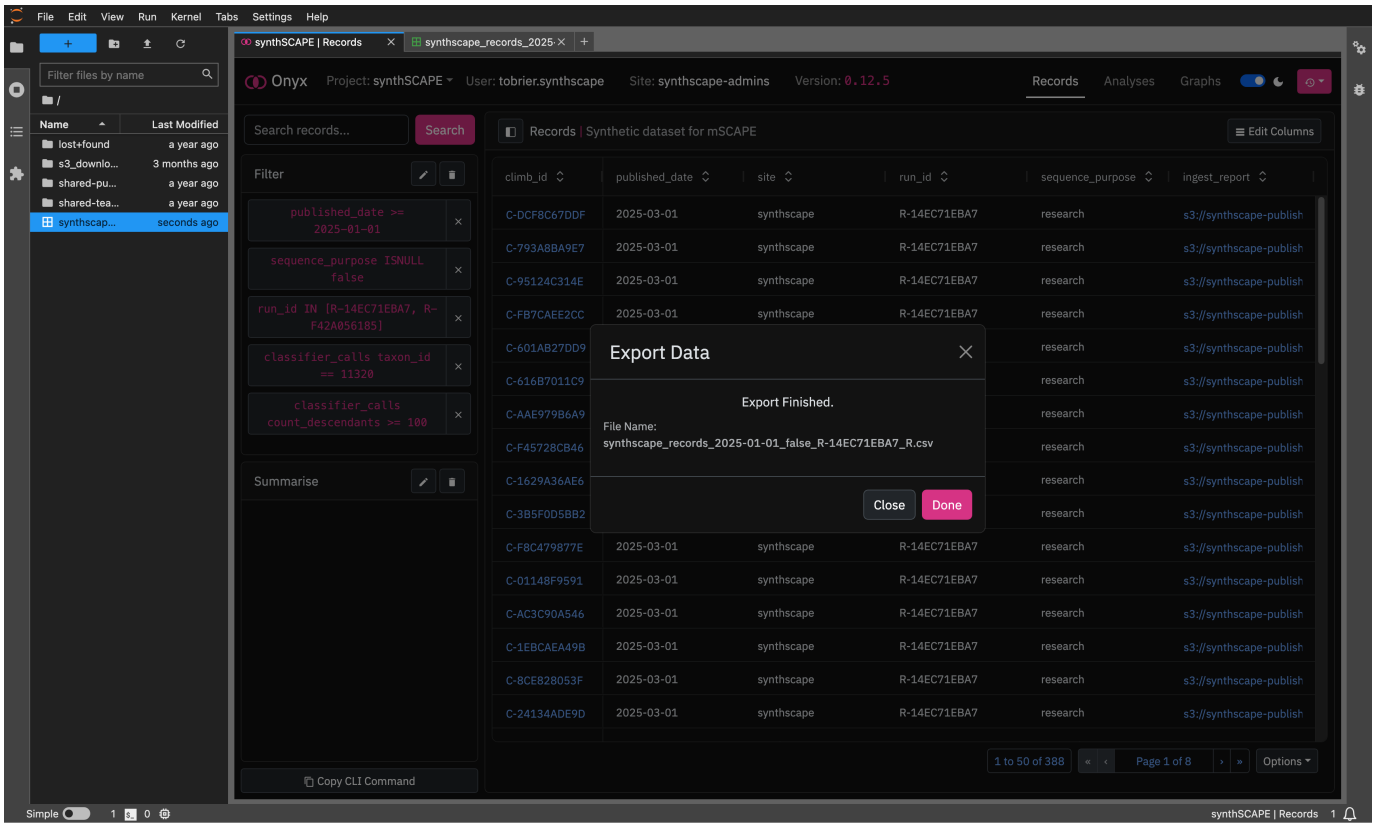


Here, we can see a default generated name for the file, and options to switch between CSV and TSV formats:



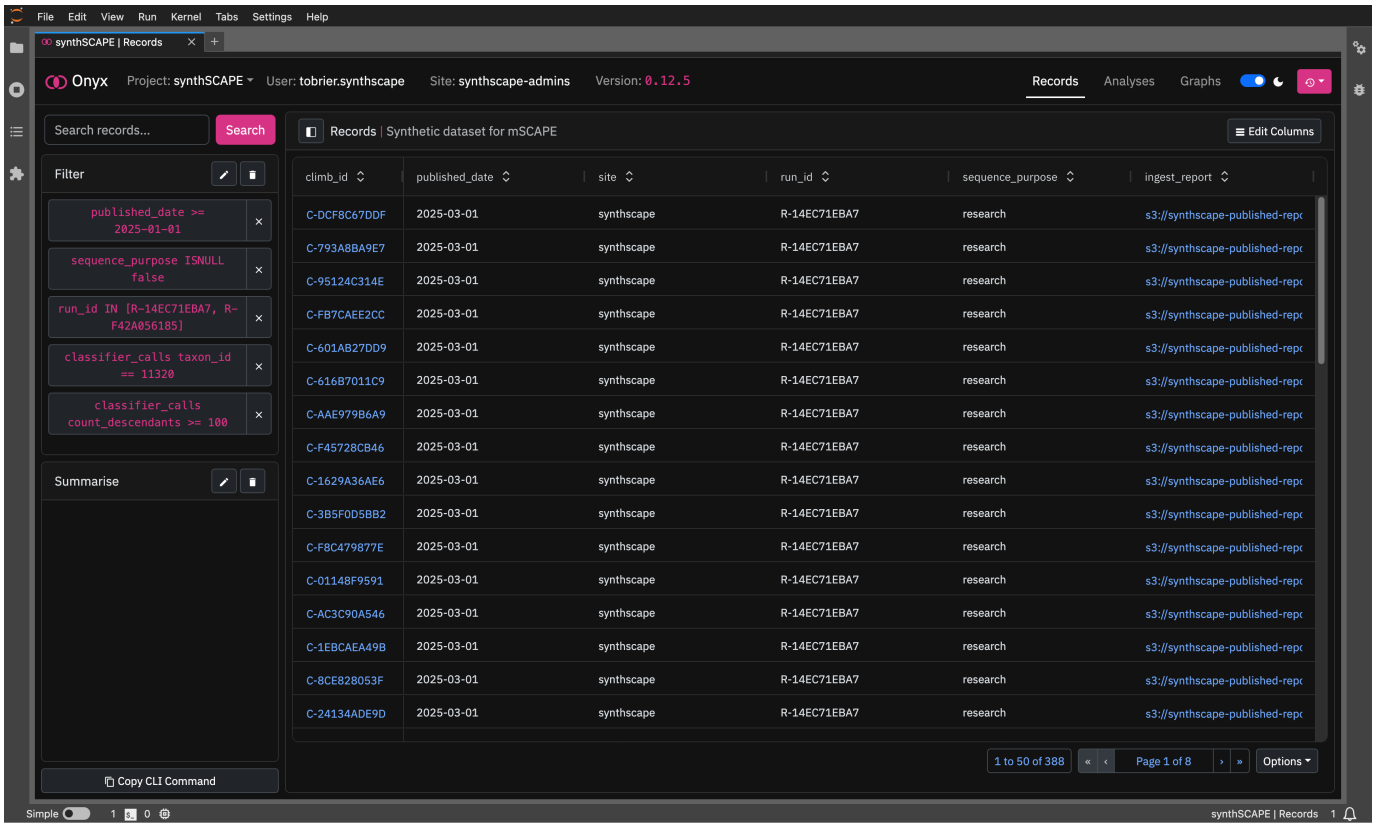
We will keep the default name and choose `.csv`.

Clicking `Export` will then pull the requested data, and create a CSV file in JupyterLab:

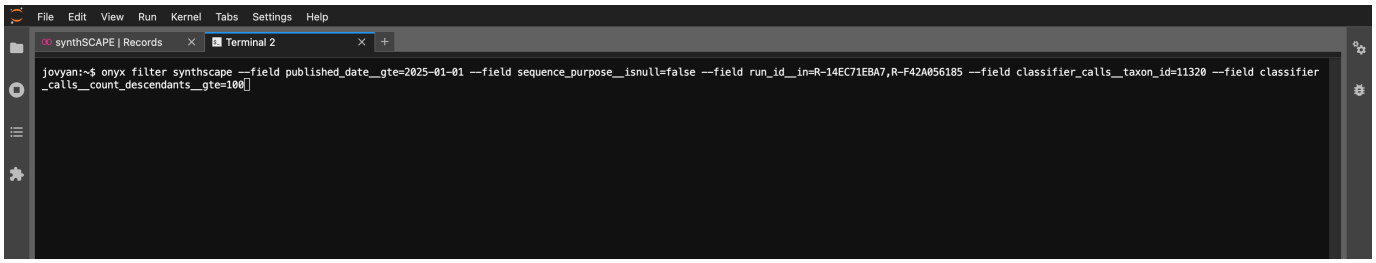


Copying to the Onyx CLI

If we wish to transfer our filters to the Onyx command-line interface, we can click `Copy CLI Command`:



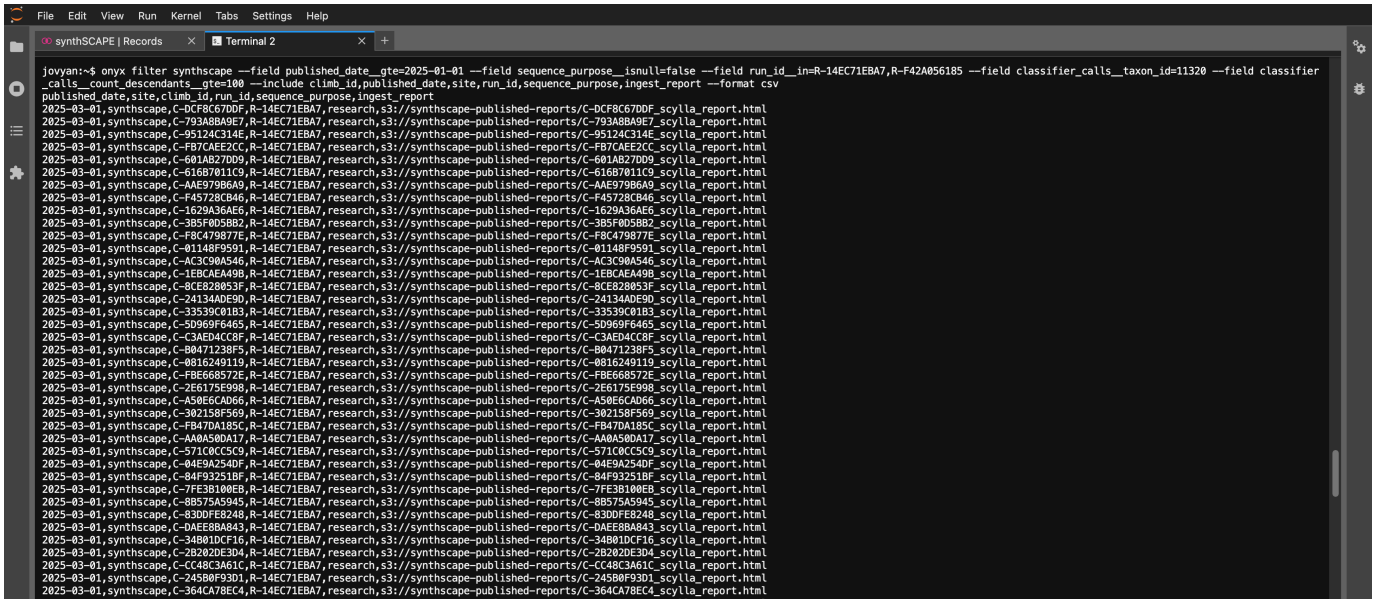
Then paste this into a terminal:



Note

`Copy CLI Command` only copies filters and summary fields; it does **not** transfer column and format settings.

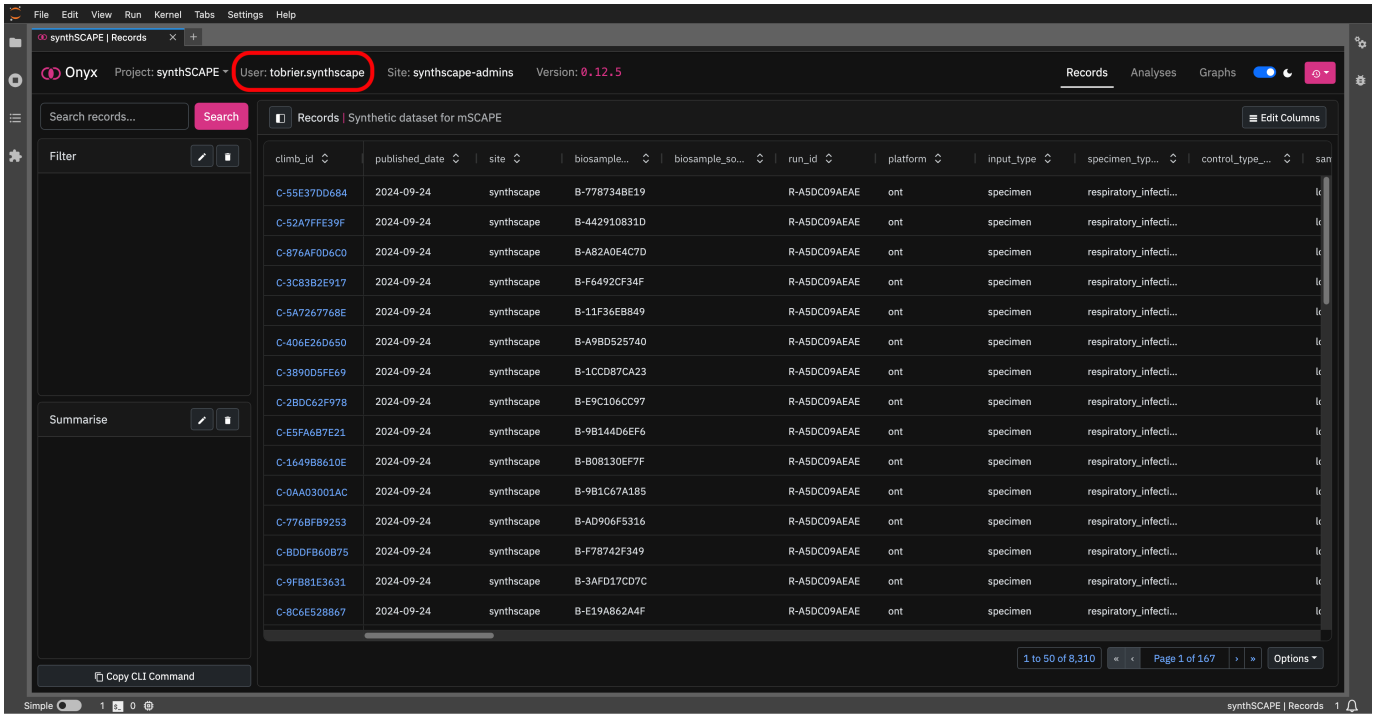
We can edit the fields/format with the `--include` and `--format` arguments, and filter the dataset:



1.3.3 Profiles & Sites

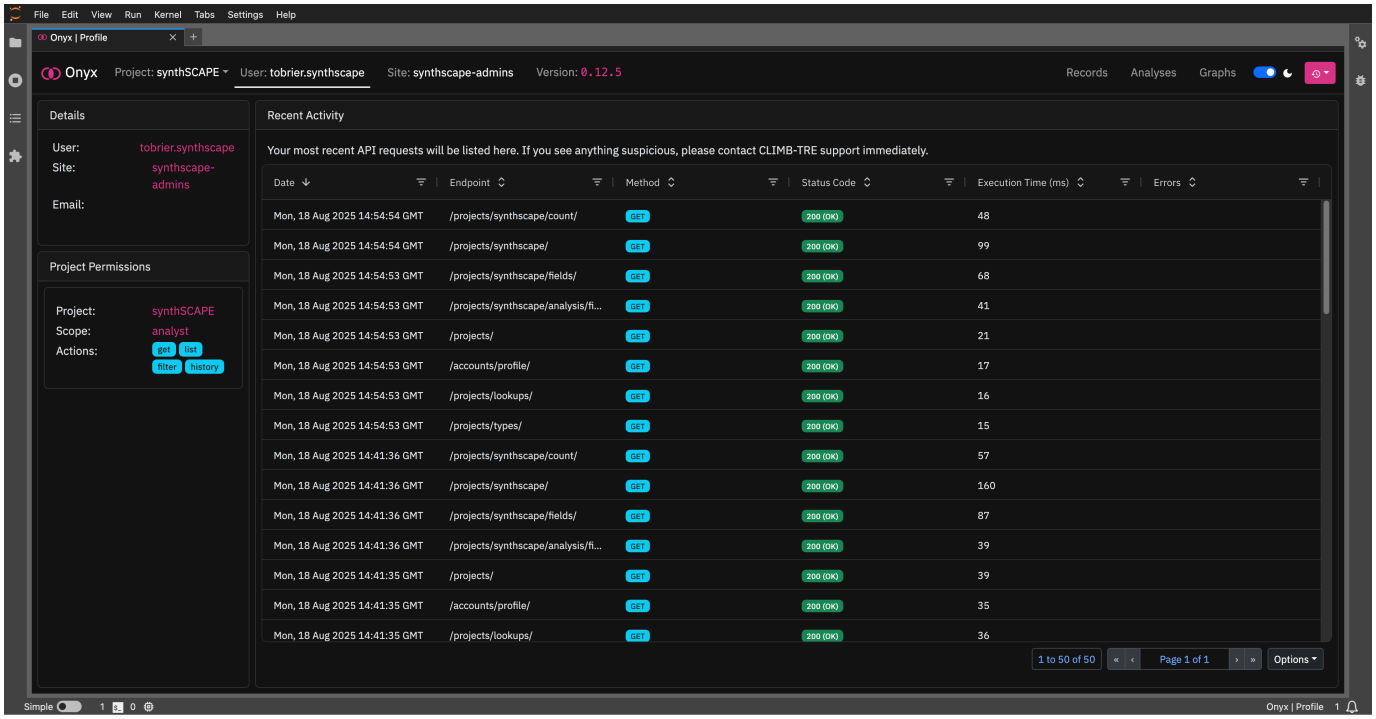
Profile Page

To view more about your profile, click on the `User: <your-username>` header:



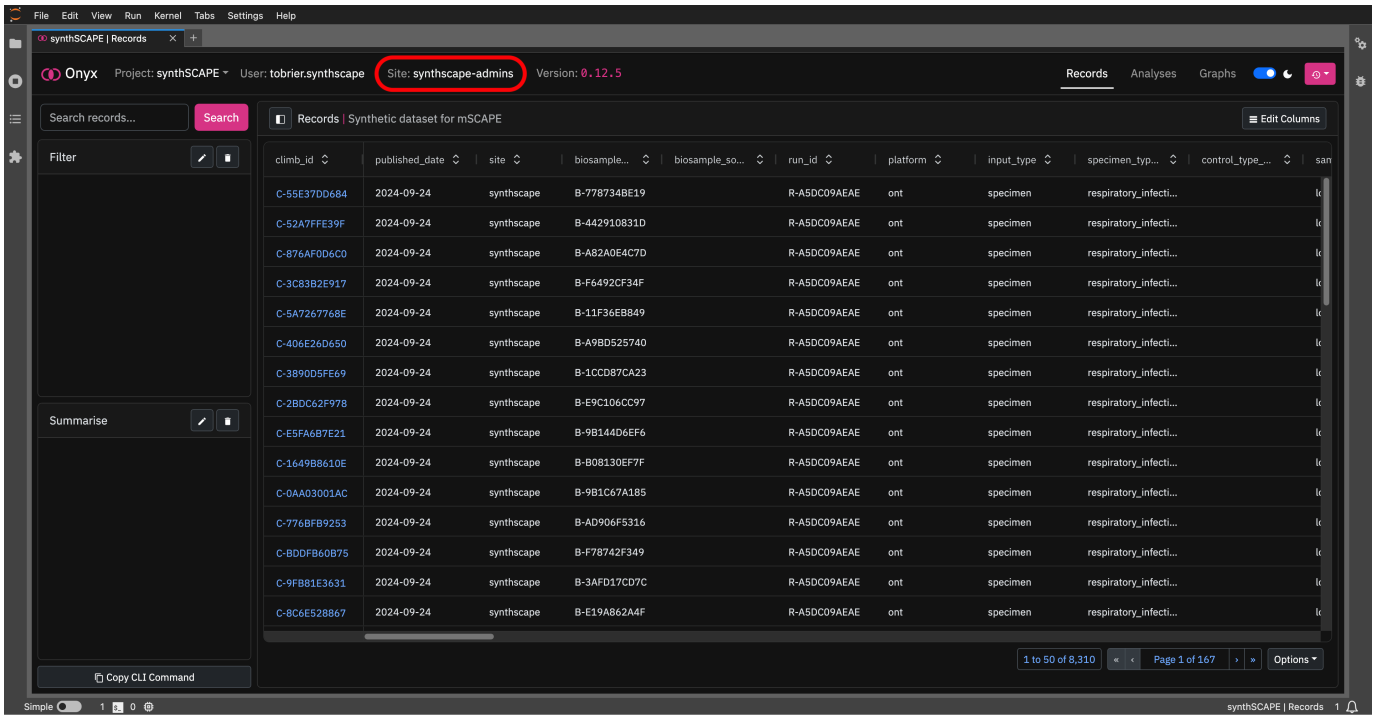
This will open the profile page, where you can see:

- `Details`: Key details about your account.
- `Project Permissions`: Projects that you have access to, and actions you can carry out on that project.
- `Recent Activity`: Your most recent 50 API requests, with status/errors/execution time information.



Site Page

To view more about your site, click on the `Site: <your-site>` header:



This will open the site page, where you can see:

- `Site Users`: Users from your site who have access to *at least* one of the same projects.