

**class** astroquery.eso.EsoClass[\[source\]](#)Bases: [QueryWithLogin](#)

User facing class to query the ESO archive

### Attributes Summary

|                                 |   |
|---------------------------------|---|
| <a href="#">AUTH_URL</a>        |   |
| <a href="#">CALSELECTOR_URL</a> |   |
| <a href="#">DOWNLOAD_URL</a>    |   |
| <a href="#">GUNZIP</a>          |   |
| <a href="#">USERNAME</a>        |   |
| <a href="#">maxrec</a>          | Getter of the maxrec attribute Safeguard that truncates the number of records returned by a query |

### Methods Summary

|  |   |
|--|---|
| <a href="#">get_associated_files</a> (datasets, *[, mode, ...])    | Invoke Calselector service to find calibration files associated to the provided datasets.                   |
| <a href="#">get_headers</a> (product_ids, *[, cache])              | Get the headers associated to a list of data product IDs  |
| <a href="#">list_instruments</a> ([cache])                         | List all the available instrument-specific queries offered by the ESO archive.                              |
| <a href="#">list_surveys</a> (*[, cache])                          | List all the available surveys (phase 3) in the ESO archive.  |
| <a href="#">login</a> (*args[, username, store_password, ...])     | Login to the ESO User Portal.   |
| <a href="#">query_apex_quicklooks</a> ([project_id, columns, ...]) | APEX data are distributed with quicklook products identified with a different name than other ESO products. |
| <a href="#">query_instrument</a> (instrument, *[, cone_ra, ...])   | Query instrument-specific raw data contained in the ESO archive.  |
| <a href="#">query_main</a> ([instruments, cone_ra, cone_dec, ...]) | Query raw data from all instruments contained in the ESO archive.   |
| <a href="#">query_surveys</a> ([surveys, cone_ra, cone_dec, ...])  | Query survey Phase 3 data contained in the ESO archive.   |
| <a href="#">query_tap_service</a> (query_str[, authenticated])     | Query the ESO TAP service using a free ADQL string.   |
| <a href="#">retrieve_data</a> (datasets, *[, continuation, ...])   | Retrieve a list of datasets form the ESO archive.   |

### Attributes Documentation

**AUTH\_URL** = 'https://www.eso.org/sso/oidc/token'**CALSELECTOR\_URL** = 'https://archive.eso.org/calselector/v1/associations'**DOWNLOAD\_URL** = 'https://dataportal.eso.org/dataPortal/file/'**GUNZIP** = 'gunzip'

**USERNAME** = ''

## maxrec

Getter of the maxrec attribute Safeguard that truncates the number of records returned by a query

## Methods Documentation

**get\_associated\_files**(*datasets*: [List\[str\]](#), \*, *mode*: [str](#) = 'raw', *savexml*: [bool](#) = False, *destination*: [str](#) | None = None) → [List\[str\]](#) [\[source\]](#)

Invoke Calselector service to find calibration files associated to the provided datasets.

**Parameters:** **datasets** : *list of strings*

List of datasets for which calibration files should be retrieved.

**mode** : *string*

Calselector mode: 'raw' (default) for raw calibrations,  
or 'processed' for processed calibrations.

**savexml** : *bool*

If true, save to disk the XML association tree returned by Calselector.

**destination** : *string*

Directory where the XML files are saved (default = astropy cache).

**Returns:** *files*

List of unique datasets associated to the input datasets.

**get\_headers**(*product\_ids*, \*, *cache*=True) [\[source\]](#)

Get the headers associated to a list of data product IDs

This method returns a [Table](#) where the rows correspond to the provided data product IDs, and the columns are from each of the Fits headers keywords.

Note: The additional column `'DP.ID'` found in the returned table corresponds to the provided data product IDs.

**Parameters:** **product\_ids** : *either a list of strings or a [Column](#)*

List of data product IDs.

**cache** : *bool*

Defaults to True. If set overrides global caching behavior. See [caching documentation](#).

**Returns:** **result** : [Table](#)

A table where: columns are header keywords, rows are product\_ids.

**list\_instruments**(*cache*=True) → [List\[str\]](#) [\[source\]](#)

List all the available instrument-specific queries offered by the ESO archive.

**Returns:** **instrument\_list** : *list of strings*

**cache** : *bool*

Deprecated - unused.

**list\_surveys**(\*, *cache*=True) → [List\[str\]](#) [\[source\]](#)

List all the available surveys (phase 3) in the ESO archive.

**Returns:** `collection_list` : *list of strings*

`cache` : *bool*

Deprecated - unused.

**login**(\*args, username: *str* = None, store\_password: *bool* = False, reenter\_password: *bool* = False, \*\*kwargs) → *bool*

Login to the ESO User Portal.

**Parameters:** `username` : *str, optional*

Username to the ESO Public Portal. If not given, it should be specified in the config file.

`store_password` : *bool, optional*

Stores the password securely in your keyring. Default is **False**.

`reenter_password` : *bool, optional*

Asks for the password even if it is already stored in the keyring. This is the way to overwrite an already stored password on the keyring. Default is **False**.

**query\_apex\_quicklooks**(project\_id: *List[str] | str* = None, \*, columns: *List | str* = None, column\_filters: *dict | None* = None, top: *int* = None, count\_only: *bool* = False, query\_str\_only: *bool* = False, help: *bool* = False, authenticated: *bool* = False, open\_form: *bool* = False, cache: *bool* = False) → *Table | int | str* [\[source\]](#)

APEX data are distributed with quicklook products identified with a different name than other ESO products. This query tool searches by project ID or any other supported keywords.

- Parameters:**
- **project\_id** (*str*) – ID of the project from which apex quicklooks data is to be queried.
  - **columns** (*str* or *list* of *str*) – Name of the columns the query should return. If specified as a string, it should be a comma-separated list of column names.
  - **top** (*int*) – When set to `N`, returns only the top `N` records.
  - **count\_only** (*bool*) – If **True**, returns only an *int*: the count of the records the query would return when set to **False**. Default is **False**.
  - **query\_str\_only** (*bool*) – If **True**, returns only a *str*: the query string that would be issued to the TAP service. Default is **False**.
  - **help** (*bool*) – If **True**, prints all the parameters accepted in `column_filters` and `columns`. Default is **False**.
  - **authenticated** (*bool*) – If **True**, runs the query as an authenticated user. Authentication must be done beforehand via **login()**. Note that authenticated queries are slower. Default is **False**.
  - **column\_filters** (*dict*, **None**) – Constraints applied to the query in ADQL syntax, `{"column_name": "<adql_operator> <value>"}`. For example: `{"exp_start": "between '2024-12-31' and '2025-12-31'"}`. Default is **None**.
  - **open\_form** (*bool*) – **Deprecated** - unused.
  - **cache** (*bool*) – **Deprecated** - unused.

- Returns:**
- By default, a **Table** containing records based on the specified columns and constraints. Returns **None** when the query has no results.
  - When `count_only` is **True**, returns an *int* representing the record count for the specified filters.
  - When `query_str_only` is **True**, returns the query string that would be issued to the TAP service given the specified arguments.

**Return type:** **Table**, *str*, *int*, or **None**

**query\_instrument**(instrument: *str*, \*, cone\_ra: *float* = None, cone\_dec: *float* = None, cone\_radius: *float* = None, columns: *List | str* = None, column\_filters: *dict | None* = None, top: *int* = None, count\_only: *bool* = False, query\_str\_only: *bool* = False, help: *bool* = False, authenticated: *bool* = False, open\_form: *bool* = False, cache: *bool* =

`False) → Table | int | str`

[\[source\]](#)

Query instrument-specific raw data contained in the ESO archive.

- Parameters:**
- **instrument** – Name of the instrument from which raw data is to be queried. Should be ONLY ONE of the names returned by `list_instruments()`.
  - **cone\_ra** – Cone Search Center - Right Ascension in degrees.
  - **cone\_dec** – Cone Search Center - Declination in degrees.
  - **cone\_radius** – Cone Search Radius in degrees.
  - **columns** (*str or list of str*) – Name of the columns the query should return. If specified as a string, it should be a comma-separated list of column names.
  - **top** (*int*) – When set to `N`, returns only the top `N` records.
  - **count\_only** (*bool*) – If **True**, returns only an **int**: the count of the records the query would return when set to **False**. Default is **False**.
  - **query\_str\_only** (*bool*) – If **True**, returns only a **str**: the query string that would be issued to the TAP service. Default is **False**.
  - **help** (*bool*) – If **True**, prints all the parameters accepted in `column_filters` and `columns`. Default is **False**.
  - **authenticated** (*bool*) – If **True**, runs the query as an authenticated user. Authentication must be done beforehand via `login()`. Note that authenticated queries are slower. Default is **False**.
  - **column\_filters** (dict, **None**) – Constraints applied to the query in ADQL syntax, `{"column_name": "<adql_operator> <value>"}`. For example: `{"exp_start": "between '2024-12-31' and '2025-12-31'"}`. Default is **None**.
  - **open\_form** (*bool*) – **Deprecated** - unused.
  - **cache** (*bool*) – **Deprecated** - unused.
- Returns:**
- By default, a **Table** containing records based on the specified columns and constraints. Returns **None** when the query has no results.
  - When `count_only` is **True**, returns an **int** representing the record count for the specified filters.
  - When `query_str_only` is **True**, returns the query string that would be issued to the TAP service given the specified arguments.

**Return type:** **Table**, **str**, **int**, or **None**

```
query_main(instruments: List[str] | str = None, *, cone_ra: float = None, cone_dec:
float = None, cone_radius: float = None, columns: List | str = None, column_filters:
dict | None = None, top: int = None, count_only: bool = False, query_str_only: bool =
False, help: bool = False, authenticated: bool = False, open_form: bool = False, cache:
bool = False) → Table | int | str
```

[\[source\]](#)

Query raw data from all instruments contained in the ESO archive.

- Parameters:**
- **instruments** (*str* or *list*) – Name of the instruments to filter. Should be one or more of the names returned by `list_instruments()`. If specified as a string, it should be a comma-separated list of instrument names. If not specified, returns records relative to all instruments. Default is **None**.
  - **cone\_ra** – Cone Search Center - Right Ascension in degrees.
  - **cone\_dec** – Cone Search Center - Declination in degrees.
  - **cone\_radius** – Cone Search Radius in degrees.
  - **columns** (*str* or *list* of *str*) – Name of the columns the query should return. If specified as a string, it should be a comma-separated list of column names.
  - **top** (*int*) – When set to `N`, returns only the top `N` records.
  - **count\_only** (*bool*) – If **True**, returns only an **int**: the count of the records the query would return when set to **False**. Default is **False**.
  - **query\_str\_only** (*bool*) – If **True**, returns only a **str**: the query string that would be issued to the TAP service. Default is **False**.
  - **help** (*bool*) – If **True**, prints all the parameters accepted in `column_filters` and `columns`. Default is **False**.
  - **authenticated** (*bool*) – If **True**, runs the query as an authenticated user. Authentication must be done beforehand via `login()`. Note that authenticated queries are slower. Default is **False**.
  - **column\_filters** (dict, **None**) – Constraints applied to the query in ADQL syntax, `{"column_name": "<adql_operator> <value>"}`. For example: `{"exp_start": "between '2024-12-31' and '2025-12-31'"}`. Default is **None**.
  - **open\_form** (*bool*) – **Deprecated** - unused.
  - **cache** (*bool*) – **Deprecated** - unused.

- Returns:**
- By default, a **Table** containing records based on the specified columns and constraints. Returns **None** when the query has no results.
  - When `count_only` is **True**, returns an **int** representing the record count for the specified filters.
  - When `query_str_only` is **True**, returns the query string that would be issued to the TAP service given the specified arguments.

**Return type:** **Table**, **str**, **int**, or **None**

```
query_surveys(surveys: List[str] | str = None, *, cone_ra: float = None, cone_dec:
float = None, cone_radius: float = None, columns: List | str = None, column_filters:
dict | None = None, top: int = None, count_only: bool = False, query_str_only: bool =
False, help: bool = False, authenticated: bool = False, open_form: bool = False, cache:
bool = False) → Table | int | str
```

[\[source\]](#)

Query survey Phase 3 data contained in the ESO archive.

- Parameters:**
- **surveys** (*str* or *list*) – Name of the survey(s) to query. Should be one or more of the names returned by `list_surveys()`. If specified as a string, it should be a comma-separated list of survey names. If not specified, returns records relative to all surveys. Default is **None**.
  - **cone\_ra** – Cone Search Center - Right Ascension in degrees.
  - **cone\_dec** – Cone Search Center - Declination in degrees.
  - **cone\_radius** – Cone Search Radius in degrees.
  - **columns** (*str* or *list* of *str*) – Name of the columns the query should return. If specified as a string, it should be a comma-separated list of column names.
  - **top** (*int*) – When set to `N`, returns only the top `N` records.
  - **count\_only** (*bool*) – If **True**, returns only an **int**: the count of the records the query would return when set to **False**. Default is **False**.
  - **query\_str\_only** (*bool*) – If **True**, returns only a **str**: the query string that would be issued to the TAP service. Default is **False**.
  - **help** (*bool*) – If **True**, prints all the parameters accepted in `column_filters` and `columns`. Default is **False**.
  - **authenticated** (*bool*) – If **True**, runs the query as an authenticated user. Authentication must be done beforehand via `login()`. Note that authenticated queries are slower. Default is **False**.
  - **column\_filters** (dict, **None**) – Constraints applied to the query in ADQL syntax, `{"column_name": "<adql_operator> <value>"}`. For example: `{"exp_start": "between '2024-12-31' and '2025-12-31'"}`. Default is **None**.
  - **open\_form** (*bool*) – **Deprecated** - unused.
  - **cache** (*bool*) – **Deprecated** - unused.
- Returns:**
- By default, a **Table** containing records based on the specified columns and constraints. Returns **None** when the query has no results.
  - When `count_only` is **True**, returns an **int** representing the record count for the specified filters.
  - When `query_str_only` is **True**, returns the query string that would be issued to the TAP service given the specified arguments.
- Return type:** **Table**, **str**, **int**, or **None**

**query\_tap\_service**(*query\_str: str, authenticated: bool = False*) → **Table** | **None** [\[source\]](#)

Query the ESO TAP service using a free ADQL string.

- Parameters:**
- **query\_str** (*str*) – The ADQL query string to be executed.
  - **authenticated** (*bool*) – If **True**, the query is run as an authenticated user. Authentication must be done beforehand via `login()`. Note that authenticated queries are slower. Default is **False**.

**Returns:** The query results in an **Table**, or **None** if no data is found.

**Return type:** Optional[**Table**]

**Example usage:**

```
eso_instance = Eso()

eso_instance.query_tap_service("SELECT * FROM ivoa.ObsCore")
```

**retrieve\_data**(*datasets, \*, continuation=False, destination=None, with\_calib=None, unzip=True, request\_all\_objects=None, request\_id=None*) [\[source\]](#)

Retrieve a list of datasets from the ESO archive.

**Parameters:** **datasets** : *list of strings or string*

List of datasets strings to retrieve from the archive.

**destination:** **string**

Directory where the files are copied. Files already found in the destination directory are skipped, unless `continuation=True`. Default to `astroquery` cache.

**continuation** : *bool*

Force the retrieval of data that are present in the destination directory.

**with\_calib** : *string*

Retrieve associated calibration files: `None` (default), `'raw'` for raw calibrations, or `'processed'` for processed calibrations.

**unzip** : *bool*

Unzip compressed files from the archive after download. **True** by default.

**Returns:** **files** : *list of strings or string*

List of files that have been locally downloaded from the archive.

## Examples

```
>>> dptbl = Eso.query_instrument('apex', pi_coi='ginsburg')
>>> dpids = [row['DP.ID'] for row in dptbl if 'Map' in row['Object']]
>>> files = Eso.retrieve_data(dpids)
```

[>>>](#)