# Package 'CuratedAtlasQueryR'

February 16, 2026

**Type** Package

**Title** Queries the Human Cell Atlas

**Version** 1.8.0

**Description** Provides access to a copy of the Human Cell Atlas, but with harmonised metadata. This allows for uniform querying across numerous datasets within the Atlas using common fields such as cell type, tissue type, and patient ethnicity. Usage involves first querying the metadata table for cells of interest, and then downloading the corresponding cells into a SingleCellExperiment object.

**License** GPL-3

**Depends** R (>= 4.2.0)

**Imports** dplyr, SummarizedExperiment, SingleCellExperiment, purrr (>= 1.0.0), BiocGenerics, glue, HDF5Array, DBI, tools, httr, cli, assertthat, SeuratObject, Seurat, methods, rlang, stats, S4Vectors, tibble, utils, dbplyr (>= 2.3.0), duckdb, stringr

**Suggests** zellkonverter, rmarkdown, knitr, testthat, basilisk, arrow, reticulate, spelling, forcats, ggplot2, tidySingleCellExperiment, rprojroot

**Biarch** true

**biocViews** AssayDomain, Infrastructure, RNASeq, DifferentialExpression, GeneExpression, Normalization, Clustering, QualityControl, Sequencing, Transcription, Transcriptomics

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**LazyDataCompression** xz

**URL** https://github.com/stemangiola/CuratedAtlasQueryR

**BugReports** https://github.com/stemangiola/CuratedAtlasQueryR/issues

**VignetteBuilder** knitr

**Roxygen** list(markdown = TRUE)

**Collate** 'utils.R' 'counts.R' 'dev.R' 'metadata.R' 'seurat.R' 'unharmonised.R' 'zzz.R'

**Language** en-US

**git_url** https://git.bioconductor.org/packages/CuratedAtlasQueryR

**git_branch** RELEASE_3_22

**git_last_commit** d6fb5df

**git_last_commit_date** 2025-10-29

**Repository** Bioconductor 3.22

**Date/Publication** 2026-02-16

**Author** Stefano Mangiola [aut, cre, rev] (ORCID:
<https://orcid.org/0000-0001-7474-836X>),
Michael Milton [aut, rev] (ORCID:
<https://orcid.org/0000-0002-8965-2595>),
Martin Morgan [ctb, rev],
Vincent Carey [ctb, rev],
Julie Iskander [rev],
Tony Papenfuss [rev],
Silicon Valley Foundation CZF2019-002443 [fnd],
NIH NHGRI 5U24HG004059-18 [fnd],
Victoria Cancer Agency ECRF21036 [fnd],
NHMRC 1116955 [fnd]

**Maintainer** Stefano Mangiola <mangiolastefano@gmail.com>

# Contents

DATABASE_URL . . . . . 2
get_default_cache_dir . . . . . 3
get_metadata . . . . . 3
get_seurat . . . . . 5
get_SingleCellExperiment . . . . . 6
get_single_cell_experiment . . . . . 7
get_unharmonised_dataset . . . . . 8
get_unharmonised_metadata . . . . . 9
hdf5_to_anndata . . . . . 10
read_parquet . . . . . 10
report_file_sizes . . . . . 11
SAMPLE_DATABASE_URL . . . . . 11
single_line_str . . . . . 12
sync_remote_file . . . . . 12
update_database . . . . . 12
update_unharmonised . . . . . 13
upload_swift . . . . . 14
url_file_size . . . . . 15

**Index** **16**

---

DATABASE_URL *URL pointing to the full metadata file*

---

## Description

URL pointing to the full metadata file

### Usage

```
DATABASE_URL
```

### Format

An object of class `character` of length 1.

### Value

A character scalar consisting of the URL

### Examples

```
get_metadata(remote_url = DATABASE_URL)
```

---

## `get_default_cache_dir` *Returns the default cache directory*

---

### Description

Returns the default cache directory

### Usage

```
get_default_cache_dir()
```

### Value

A length one character vector.

---

## `get_metadata` *Gets the Curated Atlas metadata as a data frame.*

---

### Description

Downloads a parquet database of the Human Cell Atlas metadata to a local cache, and then opens it as a data frame. It can then be filtered and passed into `get_single_cell_experiment()` to obtain a `SingleCellExperiment::SingleCellExperiment`

### Usage

```
get_metadata(
  remote_url = DATABASE_URL,
  cache_directory = get_default_cache_dir(),
  use_cache = TRUE
)
```

## Arguments

| | |
|---|---|
| remote_url | Optional character vector of length 1. An HTTP URL pointing to the location of the parquet database. |
| cache_directory | Optional character vector of length 1. A file path on your local system to a directory (not a file) that will be used to store metadata.parquet |
| use_cache | Optional logical scalar. If TRUE (the default), and this function has been called before with the same parameters, then a cached reference to the table will be returned. If FALSE, a new connection will be created no matter what. |

## Details

The metadata was collected from the Bioconductor package cellxgenedp. it's vignette using_cellxgenedp provides an overview of the columns in the metadata. The data for which the column organism_name included "Homo sapiens" was collected collected from cellxgenedp.

The columns dataset_id and file_id link the datasets explorable through CuratedAtlasQueryR and cellxgenedpto the CELLxGENE portal.

Our representation, harmonises the metadata at dataset, sample and cell levels, in a unique coherent database table.

Dataset-specific columns (definitions available at cellxgene.cziscience.com) cell_count, collection_id, created_at.x, created_at.y, dataset_deployments, dataset_id, file_id, filename, filetype, is_primary_data.y, is_valid, linked_genesets, mean_genes_per_cell, name, published, published_at, revised_at, revision, s3_uri, schema_version, tombstone, updated_at.x, updated_at.y, user_submitted, x_normalization

Sample-specific columns (definitions available at cellxgene.cziscience.com)

sample_, .sample_name, age_days, assay, assay_ontology_term_id, development_stage, development_stage_on ethnicity, ethnicity_ontology_term_id, experiment___, organism, organism_ontology_term_id, sample_placeholder, sex, sex_ontology_term_id, tissue, tissue_harmonised, tissue_ontology_term_id, disease, disease_ontology_term_id, is_primary_data.x

Cell-specific columns (definitions available at cellxgene.cziscience.com)

cell_, cell_type, cell_type_ontology_term_idm, cell_type_harmonised, confidence_class, cell_annotation_azimuth_l2, cell_annotation_blueprint_singler

Through harmonisation and curation we introduced custom column, not present in the original CELLxGENE metadata

- tissue_harmonised: a coarser tissue name for better filtering
- age_days: the number of days corresponding to the age
- cell_type_harmonised: the consensus call identity (for immune cells) using the original and three novel annotations using Seurat Azimuth and SingleR
- confidence_class: an ordinal class of how confident cell_type_harmonised is. 1 is complete consensus, 2 is 3 out of four and so on.
- cell_annotation_azimuth_l2: Azimuth cell annotation
- cell_annotation_blueprint_singler: SingleR cell annotation using Blueprint reference
- cell_annotation_blueprint_monaco: SingleR cell annotation using Monaco reference
- sample_id_db: Sample subdivision for internal use
- file_id_db: File subdivision for internal use
- sample_: Sample ID

- `.sample_name`: How samples were defined

**Possible cache path issues**

If your default R cache path includes non-standard characters (e.g. dash because of your user or organisation name), the following error can manifest

Error in `db_query_fields.DBIConnection()`: ! Can't query fields. Caused by error: ! Parser Error: syntax error at or near "/" LINE 2: FROM /Users/bob/Library/Caches...

The solution is to choose a different cache, for example

get_metadata(cache_directory = path.expand('~'))

## Value

A lazy data.frame subclass containing the metadata. You can interact with this object using most standard dplyr functions. For string matching, it is recommended that you use `stringr::str_like` to filter character columns, as `stringr::str_match` will not work.

## Examples

```
library(dplyr)
filtered_metadata <- get_metadata() |>
    filter(
        ethnicity == "African" &
            assay %LIKE% "%10x%" &
            tissue == "lung parenchyma" &
            cell_type %LIKE% "%CD4%"
    )
```

---

# get_seurat *Given a data frame of HCA metadata, returns a Seurat object corresponding to the samples in that data frame*

---

## Description

Given a data frame of HCA metadata, returns a Seurat object corresponding to the samples in that data frame

## Usage

```
get_seurat(...)
```

## Arguments

| | |
|---|---|
| `...` | Arguments passed on to `get_single_cell_experiment` |
| `data` | A data frame containing, at minimum, a `sample_` column, which corresponds to a single cell sample ID. This can be obtained from the `get_metadata()` function. |
| `assays` | A character vector whose elements must be either "counts" and/or "cpm", representing the corresponding assay(s) you want to request. By default only the count assay is downloaded. If you are interested in comparing a limited amount of genes, the "cpm" assay is more appropriate. |

repository A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.

cache_directory An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.

features An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned.

## Value

A Seurat object containing the same data as a call to get_single_cell_experiment()

## Examples

```
meta <- get_metadata() |> head(2)
seurat <- get_seurat(meta)
```

---

# get_SingleCellExperiment

*Gets a SingleCellExperiment from curated metadata*

---

## Description

Given a data frame of Curated Atlas metadata obtained from get_metadata(), returns a SingleCellExperiment::Sing object corresponding to the samples in that data frame

## Usage

```
get_SingleCellExperiment(...)
```

## Arguments

... Arguments passed on to get_single_cell_experiment

data A data frame containing, at minimum, a sample_ column, which corresponds to a single cell sample ID. This can be obtained from the get_metadata() function.

assays A character vector whose elements must be either "counts" and/or "cpm", representing the corresponding assay(s) you want to request. By default only the count assay is downloaded. If you are interested in comparing a limited amount of genes, the "cpm" assay is more appropriate.

repository A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.

cache_directory An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.

features An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned.

## Value

A SingleCellExperiment object, with one assay for each value in the assays argument

## Examples

```
meta <- get_metadata() |> head(2)
sce <- get_single_cell_experiment(meta)
```

---

`get_single_cell_experiment` *Gets a SingleCellExperiment from curated metadata*

---

## Description

Given a data frame of Curated Atlas metadata obtained from `get_metadata()`, returns a `SingleCellExperiment::Sing` object corresponding to the samples in that data frame

## Usage

```
get_single_cell_experiment(
  data,
  assays = "counts",
  cache_directory = get_default_cache_dir(),
  repository = COUNTS_URL,
  features = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | A data frame containing, at minimum, a `sample_` column, which corresponds to a single cell sample ID. This can be obtained from the `get_metadata()` function. |
| `assays` | A character vector whose elements must be either "counts" and/or "cpm", representing the corresponding assay(s) you want to request. By default only the count assay is downloaded. If you are interested in comparing a limited amount of genes, the "cpm" assay is more appropriate. |
| `cache_directory` | An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied. |
| `repository` | A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored. |
| `features` | An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned. |

## Value

A SingleCellExperiment object, with one assay for each value in the assays argument

## Examples

```
meta <- get_metadata() |> head(2)
sce <- get_single_cell_experiment(meta)
```

`get_unharmonised_dataset` *Returns unharmonised metadata for selected datasets.*

## Description

Various metadata fields are *not* common between datasets, so it does not make sense for these to live in the main metadata table. This function is a utility that allows easy fetching of this data if necessary.

## Usage

```
get_unharmonised_dataset(
  dataset_id,
  cells = NULL,
  conn = dbConnect(drv = duckdb(), read_only = TRUE),
  remote_url = UNHARMONISED_URL,
  cache_directory = get_default_cache_dir()
)
```

## Arguments

| | |
|---|---|
| `dataset_id` | A character vector, where each entry is a dataset ID obtained from the `$file_id` column of the table returned from `get_metadata()` |
| `cells` | An optional character vector of cell IDs. If provided, only metadata for those cells will be returned. |
| `conn` | An optional DuckDB connection object. If provided, it will re-use the existing connection instead of opening a new one. |
| `remote_url` | Optional character vector of length 1. An HTTP URL pointing to the root URL under which all the unharmonised dataset files are located. |
| `cache_directory` | Optional character vector of length 1. A file path on your local system to a directory (not a file) that will be used to store the unharmonised metadata files. |

## Value

A named list, where each name is a dataset file ID, and each value is a "lazy data frame", ie a `tbl`.

## Examples

```
dataset <- "838ea006-2369-4e2c-b426-b2a744a2b02b"
harmonised_meta <- get_metadata() |>
    dplyr::filter(file_id == dataset) |> dplyr::collect()
unharmonised_meta <- get_unharmonised_dataset(dataset)
unharmonised_tbl <- dplyr::collect(unharmonised_meta[[dataset]])
dplyr::left_join(harmonised_meta, unharmonised_tbl, by=c("file_id", "cell_"))
```

`get_unharmonised_metadata` *Returns unharmonised metadata for a metadata query*

## Description

Various metadata fields are *not* common between datasets, so it does not make sense for these to live in the main metadata table. This function is a utility that allows easy fetching of this data if necessary.

## Usage

```
get_unharmonised_metadata(metadata, ...)
```

## Arguments

| | |
|---|---|
| `metadata` | A lazy data frame obtained from `get_metadata()`, filtered down to some cells of interest |
| `...` | Arguments passed on to `get_unharmonised_dataset` |
| | `dataset_id` A character vector, where each entry is a dataset ID obtained from the `$file_id` column of the table returned from `get_metadata()` |
| | `cells` An optional character vector of cell IDs. If provided, only metadata for those cells will be returned. |
| | `conn` An optional DuckDB connection object. If provided, it will re-use the existing connection instead of opening a new one. |
| | `remote_url` Optional character vector of length 1. An HTTP URL pointing to the root URL under which all the unharmonised dataset files are located. |
| | `cache_directory` Optional character vector of length 1. A file path on your local system to a directory (not a file) that will be used to store the unharmonised metadata files. |

## Value

A tibble with two columns:

- `file_id`: the same `file_id` as the main metadata table obtained from `get_metadata()`
- `unharmonised`: a nested tibble, with one row per cell in the input `metadata`, containing unharmonised metadata

## Examples

```
harmonised <- dplyr::filter(get_metadata(), tissue == "kidney blood vessel")
unharmonised <- get_unharmonised_metadata(harmonised)
```

## hdf5_to_anndata *Converts a series of HDF5Array-serialized SingleCellExperiments to AnnData*

### Description

Converts a series of HDF5Array-serialized SingleCellExperiments to AnnData

### Usage

```
hdf5_to_anndata(input_directory, output_directory)
```

### Arguments

| | |
|---|---|
| input_directory | A character scalar. The path to a directory containing one or more directories created by HDF5Array::saveHDF5SummarizedExperiment(). |
| output_directory | A character scalar. The path to a directory in which to save the created anndata files. |

### Value

A character vector of the newly-created anndata files

### Examples

```
hdf5_to_anndata(
    "/vast/projects/cellxgene_curated/splitted_DB2_data_0.2.1",
    "/vast/projects/cellxgene_curated/splitted_DB2_anndata_0.2.1"
)
hdf5_to_anndata(
    "/vast/projects/cellxgene_curated/splitted_DB2_data_scaled_0.2.1",
    "/vast/projects/cellxgene_curated/splitted_DB2_anndata_scaled_0.2.1"
)
```

## read_parquet *Returns a tibble from a parquet file path*

### Description

Since dbplyr 2.4.0, raw file paths aren't handled very well See: https://github.com/duckdb/duckdb-r/issues/38 Hence the need for this method

### Usage

```
read_parquet(conn, path)
```

### Value

An SQL data frame

## report_file_sizes *Prints a message indicating the size of a download*

### Description

Prints a message indicating the size of a download

### Usage

```
report_file_sizes(urls)
```

### Arguments

| | |
|---|---|
| urls | A character vector containing URLs |

### Value

NULL, invisibly

## SAMPLE_DATABASE_URL *URL pointing to the sample metadata file, which is smaller and for test, demonstration, and vignette purposes only*

### Description

URL pointing to the sample metadata file, which is smaller and for test, demonstration, and vignette purposes only

### Usage

```
SAMPLE_DATABASE_URL
```

### Format

An object of class character of length 1.

### Value

A character scalar consisting of the URL

### Examples

```
get_metadata(remote_url = SAMPLE_DATABASE_URL)
```

## single_line_str *Formats a multi-line string as it it were on one line*

### Description

Formats a multi-line string as it it were on one line

### Usage

```
single_line_str(text)
```

### Arguments

| | |
|---|---|
| text | Any character vector |

### Value

The same character vector, with newlines and subsequent whitespace removed

## sync_remote_file *Synchronises a single remote file with a local path*

### Description

Synchronises a single remote file with a local path

### Usage

```
sync_remote_file(full_url, output_file, ...)
```

### Value

NULL, invisibly

## update_database *Update the metadata database in nectar using a newly created data frame*

### Description

Update the metadata database in nectar using a newly created data frame

### Usage

```
update_database(metadata, version, ...)
```

### Arguments

| | |
|---|---|
| metadata | The data frame to upload |
| version | The version for the new metadata as a character scalar, e.g. "0.2.3" |
| ... | Arguments passed on to upload_swift<br>source A character scalar indicating the local path to the file to upload<br>container A character scalar indicating the name of the container to upload to<br>name An optional character scalar indicating the name the file should have after being uploaded. Defaults to being the basename of the source file.<br>credential_id The OpenStack application credential secret as a character scalar |

### Value

NULL, invisibly

### Examples

```
## Not run:
 metadata = CuratedAtlasQueryR::get_metadata() |>
     head(10) |>
     dplyr::collect()
 update_database(
     metadata,
     "0.2.3",
     credential_id = "ABCDEFGHIJK",
     credential_secret = "ABCD1234EFGH-5678IJK"
 )
 # Prints "metadata.0.2.3.parquet" if successful

## End(Not run)
```

---

## update_unharmonised *Update the unharmonised parquet files*

---

### Description

Update the unharmonised parquet files

### Usage

```
update_unharmonised(unharmonised_parquet_dir, ...)
```

### Arguments

| | |
|---|---|
| unharmonised_parquet_dir | The path to a directory containing parquet files, one for each dataset, e.g. /vast/projects/cellxgene_cura |
| ... | Arguments passed on to upload_swift<br>source A character scalar indicating the local path to the file to upload<br>container A character scalar indicating the name of the container to upload to<br>name An optional character scalar indicating the name the file should have after being uploaded. Defaults to being the basename of the source file.<br>credential_id The OpenStack application credential secret as a character scalar |

## Value

NULL, invisibly

## Examples

```
## Not run:
update_unharmonised(
    "/vast/projects/cellxgene_curated/metadata_non_harmonised_parquet_0.2",
    credential_id = "ABCDEFGHIJK",
    credential_secret = "ABCD1234EFGH-5678IJK"
)

## End(Not run)
```

---

# upload_swift *Upload a file to the Nectar object store*

---

## Description

Upload a file to the Nectar object store

## Usage

```
upload_swift(
  source,
  container,
  name = basename(source),
  credential_id = NULL,
  credential_secret = NULL
)
```

## Arguments

| | |
|---|---|
| source | A character scalar indicating the local path to the file to upload |
| container | A character scalar indicating the name of the container to upload to |
| name | An optional character scalar indicating the name the file should have after being uploaded. Defaults to being the basename of the source file. |
| credential_id | The OpenStack application credential secret as a character scalar |

## Value

NULL, invisibly

`url_file_size` *Gets the file size of a number of remote files*

## Description

Gets the file size of a number of remote files

## Usage

```
url_file_size(urls)
```

## Arguments

| | |
|---|---|
| `urls` | A character vector containing URLs |

## Value

The file size of each of the files pointed to by the provided URL, in gigabytes, as double vector

# Index


∗ **datasets**
DATABASE_URL, 2
SAMPLE_DATABASE_URL, 11
∗ **internal**
get_default_cache_dir, 3
hdf5_to_anndata, 10
read_parquet, 10
report_file_sizes, 11
single_line_str, 12
sync_remote_file, 12
update_database, 12
update_unharmonised, 13
upload_swift, 14
url_file_size, 15

DATABASE_URL, 2

get_default_cache_dir, 3
get_metadata, 3
get_metadata(), *5–9*
get_seurat, 5
get_single_cell_experiment, *5, 6*, 7
get_single_cell_experiment(), *3*, *6*
get_SingleCellExperiment, 6
get_unharmonised_dataset, 8, *9*
get_unharmonised_metadata, 9

hdf5_to_anndata, 10
HDF5Array::saveHDF5SummarizedExperiment(), *10*

read_parquet, 10
report_file_sizes, 11

SAMPLE_DATABASE_URL, 11
single_line_str, 12
SingleCellExperiment::SingleCellExperiment, *3*, *6, 7*
sync_remote_file, 12

update_database, 12
update_unharmonised, 13
upload_swift, *13*, 14
url_file_size, 15