

# Package: rdataretriever (via r-universe)

November 22, 2024

**Title** R Interface to the Data Retriever

**Description** Provides an R interface to the Data Retriever  
<<https://retriever.readthedocs.io/en/latest/>> via the Data  
Retriever's command line interface. The Data Retriever  
automates the tasks of finding, downloading, and cleaning  
public datasets, and then stores them in a local database.

**Version** 3.1.1

**BugReports** <https://github.com/ropensci/rdataretriever/issues>

**URL** <https://docs.ropensci.org/rdataretriever/> (website),  
<https://github.com/ropensci/rdataretriever/>

**Depends** R (>= 3.4.0)

**Imports** reticulate (>= 1.16), semver

**Suggests** testthat (>= 1.0.0), DBI, devtools, RSQLite, RPostgreSQL,  
knitr, rmarkdown, dbplyr, raster, ggplot2

**VignetteBuilder** knitr

**SystemRequirements** Python (>= 3.7), retriever (>= 3.0.0) (version must  
be listed to patch to allow parsing)

**License** MIT + file LICENSE

**RoxygenNote** 7.2.3

**Encoding** UTF-8

**Config/pak/sysreqs** libpng-dev python3

**Repository** <https://ropensci.r-universe.dev>

**RemoteUrl** <https://github.com/ropensci/rdataretriever>

**RemoteRef** main

**RemoteSha** 36d927291f1d2d8edbb41d0bd1ef0ff22de4fcfe

## Contents

check_for_updates . . . . .	2
check_retriever_availability . . . . .	3
commit . . . . .	4
commit_log . . . . .	4
datasets . . . . .	5
dataset_names . . . . .	6
data_retriever_version . . . . .	6
display_all_rdataset_names . . . . .	7
download . . . . .	7
fetch . . . . .	8
find_socrata_dataset_by_id . . . . .	9
get_dataset_names_upstream . . . . .	10
get_rdataset_names . . . . .	10
get_retriever_citation . . . . .	11
get_script_citation . . . . .	11
get_script_upstream . . . . .	12
get_updates . . . . .	12
install . . . . .	13
install_csv . . . . .	14
install_json . . . . .	15
install_msaccess . . . . .	16
install_mysql . . . . .	17
install_postgres . . . . .	18
install_retriever . . . . .	19
install_sqlite . . . . .	20
install_xml . . . . .	21
reload_scripts . . . . .	22
reset . . . . .	22
socrata_autocomplete_search . . . . .	23
socrata_dataset_info . . . . .	23
update_rdataset_catalog . . . . .	24
use_RetrievePath . . . . .	24
<b>Index</b>	<b>26</b>

---

check_for_updates	<i>Check for updates</i>
-------------------	--------------------------

---

### Description

Check for updates

### Usage

```
check_for_updates(repo = "")
```

### Arguments

repo                    path to the repository

### Value

No return value, checks for updates repo

### Examples

```
## Not run:  
rdataretriever::check_for_updates()  
  
## End(Not run)
```

---

check\_retriever\_availability

*Check to see if minimum version of retriever Python package is installed*

---

### Description

Check to see if minimum version of retriever Python package is installed

### Usage

```
check_retriever_availability()
```

### Value

boolean

### Examples

```
## Not run:  
rdataretriever::check_retriever_availability()  
  
## End(Not run)
```

---

commit	<i>Commit a dataset</i>
--------	-------------------------

---

**Description**

Commit a dataset

**Usage**

```
commit(dataset, commit_message = "", path = NULL, quiet = FALSE)
```

**Arguments**

dataset	name of the dataset
commit_message	commit message for the commit
path	path to save the committed dataset, if no path given save in provenance directory
quiet	logical, if true retriever runs in quiet mode

**Value**

No return value, provides confirmation for commit

**Examples**

```
## Not run:
rdataretriever::commit("iris")

## End(Not run)
```

---

commit_log	<i>See the log of committed dataset stored in provenance directory</i>
------------	--

---

**Description**

See the log of committed dataset stored in provenance directory

**Usage**

```
commit_log(dataset)
```

**Arguments**

dataset	name of the dataset stored in provenance directory
---------	--

**Value**

No return value, prints message after commit

**Examples**

```
## Not run:  
rdataretriever::commit_log("iris")  
  
## End(Not run)
```

---

datasets	<i>Name all available dataset scripts.</i>
----------	--

---

**Description**

Additional information on the available datasets can be found at url <https://retriever.readthedocs.io/en/latest/datasets.html>

**Usage**

```
datasets(keywords = "", licenses = "")
```

**Arguments**

keywords	search all datasets by keywords
licenses	search all datasets by licenses

**Value**

returns a character vector with the available datasets for download

Returns the names of all available dataset scripts

**Examples**

```
## Not run:  
rdataretriever::datasets()  
  
## End(Not run)
```

dataset\_names            *Name all available dataset scripts.*

---

**Description**

Additional information on the available datasets can be found at url <https://retriever.readthedocs.io/en/latest/datasets.html>

**Usage**

```
dataset_names()
```

**Value**

returns a character vector with the available datasets for download

**Examples**

```
## Not run:  
rdataretriever::dataset_names()  
  
## End(Not run)
```

---

data\_retriever\_version  
                          *Get Data Retriever version*

---

**Description**

Get Data Retriever version

**Usage**

```
data_retriever_version(clean = TRUE)
```

**Arguments**

clean                    boolean return cleaned version appropriate for semver

**Value**

returns a string with the version information

**Examples**

```
## Not run:  
rdataretriever::data_retriever_version()  
  
## End(Not run)
```

---

`display_all_rdataset_names`

*Displays the list of rdataset names present in the list of packages provided*

---

**Description**

Can take a list of packages, or NULL or a string 'all' for all rdataset packages and datasets

**Usage**

```
display_all_rdataset_names(package_name = NULL)
```

**Arguments**

`package_name` print datasets in the package, default to print rdataset and all to print all

**Value**

No return value, displays the list of rdataset names present

**Examples**

```
## Not run:  
rdataretriever::display_all_rdataset_names()  
  
## End(Not run)
```

---

`download`

*Download datasets via the Data Retriever.*

---

**Description**

Directly downloads data files with no processing, allowing downloading of non-tabular data.

**Usage**

```
download(  
  dataset,  
  path = "./",  
  quiet = FALSE,  
  sub_dir = "",  
  debug = FALSE,  
  use_cache = TRUE  
)
```

**Arguments**

dataset	the name of the dataset that you wish to download
path	the path where the data should be downloaded to
quiet	logical, if true retriever runs in quiet mode
sub_dir	downloaded dataset is stored into a custom subdirectory.
debug	setting TRUE helps in debugging in case of errors
use_cache	Setting FALSE reinstalls scripts even if they are already installed

**Value**

No return value, downloads the raw dataset

**Examples**

```
## Not run:
rdataretriever::download("plant-comp-ok")
# downloaded files will be copied to your working directory
# when no path is specified
dir()

## End(Not run)
```

---

 fetch

---

*Fetch a dataset via the Data Retriever*


---

**Description**

Each datafile in a given dataset is downloaded to a temporary directory and then imported as a data.frame as a member of a named list.

**Usage**

```
fetch(dataset, quiet = TRUE, data_names = NULL)
```

**Arguments**

dataset	the names of the dataset that you wish to download
quiet	logical, if true retriever runs in quiet mode
data_names	the names you wish to assign to cells of the list which stores the fetched dataframes. This is only relevant if you are downloading more than one dataset.

**Value**

Returns a dataframe of dataset



**Examples**

```
## Not run:
## fetch the portal Database
portal <- rdataretriever::fetch("portal")
class(portal)
names(portal)
## preview the data in the portal species datafile
head(portal$species)
vegdata <- rdataretriever::fetch(c("plant-comp-ok", "plant-occur-oosting"))
names(vegdata)
names(vegdata$plant_comp_ok)

## End(Not run)
```

---

find\_socrata\_dataset\_by\_id

*Returns metadata for the following dataset id*

---

**Description**

Returns metadata for the following dataset id

**Usage**

```
find_socrata_dataset_by_id(dataset_id)
```

**Arguments**

dataset\_id      id of the dataset

**Value**

No return value, shows metadata for the following dataset id

**Examples**

```
## Not run:
rdataretriever::socrata_dataset_info()

## End(Not run)
```

get\_dataset\_names\_upstream  
*Get dataset names from upstream*

---

**Description**

Get dataset names from upstream

**Usage**

```
get_dataset_names_upstream(keywords = "", licenses = "", repo = "")
```

**Arguments**

keywords	filter datasets based on keywords
licenses	filter datasets based on license
repo	path to the repository

**Value**

No return value, gets dataset names from upstream

**Examples**

```
## Not run:  
rdataretriever::get_dataset_names_upstream(keywords = "", licenses = "", repo = "")  
  
## End(Not run)
```

---

get\_rdataset\_names *Returns a list of all the available RDataset names present*

---

**Description**

Returns a list of all the available RDataset names present

**Usage**

```
get_rdataset_names()
```

**Value**

No return value, list of all the available RDataset

**Examples**

```
## Not run:  
rdataretriever::get_rdataset_names()  
  
## End(Not run)
```

---

*get\_retriever\_citation*      *Get retriever citation*

---

**Description**

Get retriever citation

**Usage**

```
get_retriever_citation()
```

**Value**

No return value, outputs citation of the Data Retriever Python package

**Examples**

```
## Not run:  
rdataretriever::get_retriever_citation()  
  
## End(Not run)
```

---

*get\_script\_citation*      *Get citation of a script*

---

**Description**

Get citation of a script

**Usage**

```
get_script_citation(dataset = NULL)
```

**Arguments**

dataset              dataset to obtain citation

**Value**

No return value, gets citation of a script

**Examples**

```
## Not run:
rdataretriever::get_script_citation(dataset = "")

## End(Not run)
```

---

get\_script\_upstream     *Get scripts upstream*

---

**Description**

Get scripts upstream

**Usage**

```
get_script_upstream(dataset, repo = "")
```

**Arguments**

dataset	name of the dataset
repo	path to the repository

**Value**

No return value, gets upstream scripts

**Examples**

```
## Not run:
rdataretriever::get_script_upstream("iris")

## End(Not run)
```

---

get\_updates             *Update the retriever's dataset scripts to the most recent versions.*

---

**Description**

This function will check if the version of the retriever's scripts in your local directory '`~/retriever/scripts/`' is up-to-date with the most recent official retriever release. Note it is possible that even more updated scripts exist at the retriever repository <https://github.com/weecology/retriever/tree/main/scripts> that have not yet been incorporated into an official release, and you should consider checking that page if you have any concerns.

**Usage**

```
get_updates()
```

**Value**

No return value, update the retriever's dataset scripts to the most recent versions

**Examples**

```
## Not run:
rdataretriever::get_updates()

## End(Not run)
```

---

install	<i>Install datasets via the Data Retriever (deprecated).</i>
---------	--

---

**Description**

Data is stored in either CSV files or one of the following database management systems: MySQL, PostgreSQL, SQLite, or Microsoft Access.

**Usage**

```
install(
  dataset,
  connection,
  db_file = NULL,
  conn_file = NULL,
  data_dir = ".",
  log_dir = NULL
)
```

**Arguments**

dataset	the name of the dataset that you wish to download								
connection	what type of database connection should be used. The options include: mysql, postgres, sqlite, msaccess, or csv'								
db_file	the name of the database file the dataset should be loaded into								
conn_file	the path to the .conn file that contains the connection configuration options for mysql and postgres databases. This defaults to mysql.conn or postgres.conn respectively. The connection file is a file that is formatted in the following way: <table style="margin-left: 40px;"> <tr> <td>host</td> <td>my_server@my_host.com</td> </tr> <tr> <td>port</td> <td>my_port_number</td> </tr> <tr> <td>user</td> <td>my_user_name</td> </tr> <tr> <td>password</td> <td>my_password</td> </tr> </table>	host	my_server@my_host.com	port	my_port_number	user	my_user_name	password	my_password
host	my_server@my_host.com								
port	my_port_number								
user	my_user_name								
password	my_password								
data_dir	the location where the dataset should be installed. Only relevant for csv connection types. Defaults to current working directory								
log_dir	the location where the retriever log should be stored if the progress is not printed to the console								

**Value**

No return value, main install function

**Examples**

```
## Not run:
rdataretriever::install("iris", "csv")

## End(Not run)
```

---

install\_csv

*Install datasets via the Data Retriever.*


---

**Description**

Data is stored in CSV files

**Usage**

```
install_csv(
  dataset,
  table_name = "{db}_{table}.csv",
  data_dir = getwd(),
  debug = FALSE,
  use_cache = TRUE,
  force = FALSE,
  hash_value = NULL
)
```

**Arguments**

dataset	the name of the dataset that you wish to install or path to a committed dataset zip file
table_name	the name of the database file to store data
data_dir	the dir path to store data, defaults to working dir
debug	setting TRUE helps in debugging in case of errors
use_cache	Setting FALSE reinstalls scripts even if they are already installed
force	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
hash_value	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets into CSV

**Examples**

```
## Not run:
rdataretriever::install_csv("iris")

## End(Not run)
```

---

`install_json`*Install datasets via the Data Retriever.*

---

**Description**

Data is stored in JSON files

**Usage**

```
install_json(
  dataset,
  table_name = "{db}_{table}.json",
  data_dir = getwd(),
  debug = FALSE,
  use_cache = TRUE,
  force = FALSE,
  hash_value = NULL
)
```

**Arguments**

<code>dataset</code>	the name of the dataset that you wish to install or path to a committed dataset zip file
<code>table_name</code>	the name of the database file to store data
<code>data_dir</code>	the dir path to store data, defaults to working dir
<code>debug</code>	setting TRUE helps in debugging in case of errors
<code>use_cache</code>	setting FALSE reinstalls scripts even if they are already installed
<code>force</code>	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
<code>hash_value</code>	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets in to JSON

**Examples**

```
## Not run:
rdataretriever::install_json("iris")

## End(Not run)
```

---

install\_msaccess      *Install datasets via the Data Retriever.*

---

### Description

Data is stored in MSAccess database

### Usage

```
install_msaccess(  
  dataset,  
  file = "access.mdb",  
  table_name = "[{db} {table}]",  
  debug = FALSE,  
  use_cache = TRUE,  
  force = FALSE,  
  hash_value = NULL  
)
```

### Arguments

dataset	the name of the dataset that you wish to install or path to a committed dataset zip file
file	file name for database
table_name	table name for installing of dataset
debug	setting TRUE helps in debugging in case of errors
use_cache	Setting FALSE reinstalls scripts even if they are already installed
force	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
hash_value	the hash value of committed dataset when installing from provenance directory

### Value

No return value, installs datasets into MSAccess database

### Examples

```
## Not run:  
rdat retriever::install_msaccess(dataset = "iris", file = "sqlite.db")  
  
## End(Not run)
```



---

`install_mysql`*Install datasets via the Data Retriever.*

---

**Description**

Data is stored in MySQL database

**Usage**

```
install_mysql(  
  dataset,  
  user = "root",  
  password = "",  
  host = "localhost",  
  port = 3306,  
  database_name = "{db}",  
  table_name = "{db}.{table}",  
  debug = FALSE,  
  use_cache = TRUE,  
  force = FALSE,  
  hash_value = NULL  
)
```

**Arguments**

<code>dataset</code>	the name of the dataset that you wish to install or path to a committed dataset zip file
<code>user</code>	username for database connection
<code>password</code>	password for database connection
<code>host</code>	hostname for connection
<code>port</code>	port number for connection
<code>database_name</code>	database name in which dataset will be installed
<code>table_name</code>	table name specified especially for datasets containing one file
<code>debug</code>	setting TRUE helps in debugging in case of errors
<code>use_cache</code>	setting FALSE reinstalls scripts even if they are already installed
<code>force</code>	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
<code>hash_value</code>	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets into MySQL database

**Examples**

```
## Not run:
rdatretriever::install_mysql(dataset = "portal", user = "postgres", password = "abcdef")

## End(Not run)
```

---

install\_postgres      *Install datasets via the Data Retriever.*

---

**Description**

Data is stored in PostgreSQL database

**Usage**

```
install_postgres(
  dataset,
  user = "postgres",
  password = "",
  host = "localhost",
  port = 5432,
  database = "postgres",
  database_name = "{db}",
  table_name = "{db}.{table}",
  bbox = list(),
  debug = FALSE,
  use_cache = TRUE,
  force = FALSE,
  hash_value = NULL
)
```

**Arguments**

dataset	the name of the dataset that you wish to install or path to a committed dataset zip file
user	username for database connection
password	password for database connection
host	hostname for connection
port	port number for connection
database	the database name default is postgres
database_name	database schema name in which dataset will be installed
table_name	table name specified especially for datasets containing one file
bbox	optional extent values used to fetch data from the spatial dataset
debug	setting TRUE helps in debugging in case of errors

use_cache	setting FALSE reinstalls scripts even if they are already installed
force	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
hash_value	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets into PostgreSQL database

**Examples**

```
## Not run:
rdataretriever::install_postgres(dataset = "portal", user = "postgres", password = "abcdef")

## End(Not run)
```

---

install_retriever	<i>install the python module 'retriever'</i>
-------------------	--

---

**Description**

install the python module 'retriever'

**Usage**

```
install_retriever(method = "auto", conda = "auto")
```

**Arguments**

method	Installation method. By default, "auto" automatically finds a method that will work in the local environment. Change the default to force a specific installation method. Note that the "virtualenv" method is not available on Windows.
conda	The path to a conda executable. Use "auto" to allow reticulate to automatically find an appropriate conda binary. See <b>Finding Conda</b> and <a href="#">conda_binary()</a> for more details.

**Value**

No return value, install the python module 'retriever'

---

`install_sqlite`*Install datasets via the Data Retriever.*

---

**Description**

Data is stored in SQLite database

**Usage**

```
install_sqlite(  
  dataset,  
  file = "sqlite.db",  
  table_name = "{db}_{table}",  
  data_dir = getwd(),  
  debug = FALSE,  
  use_cache = TRUE,  
  force = FALSE,  
  hash_value = NULL  
)
```

**Arguments**

<code>dataset</code>	the name of the dataset that you wish to install or path to a committed dataset zip file
<code>file</code>	SQLite database file name or path
<code>table_name</code>	table name for installing of dataset
<code>data_dir</code>	the dir path to store the db, defaults to working dir
<code>debug</code>	setting TRUE helps in debugging in case of errors
<code>use_cache</code>	setting FALSE reinstalls scripts even if they are already installed
<code>force</code>	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
<code>hash_value</code>	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets into SQLite database

**Examples**

```
## Not run:  
rdataretriever::install_sqlite(dataset = "iris", file = "sqlite.db")  
  
## End(Not run)
```

---

`install_xml`*Install datasets via the Data Retriever.*

---

**Description**

Data is stored in XML files

**Usage**

```
install_xml(  
  dataset,  
  table_name = "{db}_{table}.xml",  
  data_dir = getwd(),  
  debug = FALSE,  
  use_cache = TRUE,  
  force = FALSE,  
  hash_value = NULL  
)
```

**Arguments**

<code>dataset</code>	the name of the dataset that you wish to install or path to a committed dataset zip file
<code>table_name</code>	the name of the database file to store data
<code>data_dir</code>	the dir path to store data, defaults to working dir
<code>debug</code>	setting TRUE helps in debugging in case of errors
<code>use_cache</code>	Setting FALSE reinstalls scripts even if they are already installed
<code>force</code>	setting TRUE doesn't prompt for confirmation while installing committed datasets when changes are discovered in environment
<code>hash_value</code>	the hash value of committed dataset when installing from provenance directory

**Value**

No return value, installs datasets into XML

**Examples**

```
## Not run:  
rdataretriever::install_xml("iris")  
  
## End(Not run)
```

---

reload_scripts	<i>Update the retriever's global_script_list with the scripts present in the ~/.retriever directory.</i>
----------------	--

---

**Description**

Update the retriever's global\_script\_list with the scripts present in the ~/.retriever directory.

**Usage**

```
reload_scripts()
```

**Value**

No return value, fetches most resent scripts

**Examples**

```
## Not run:  
rdataretriever::reload_scripts()  
  
## End(Not run)
```

---

reset	<i>Reset the scripts or data(raw_data) directory or both</i>
-------	--

---

**Description**

Reset the scripts or data(raw\_data) directory or both

**Usage**

```
reset(scope = "all")
```

**Arguments**

scope	All resets both scripst and data directory
-------	--

**Value**

No return value, resets the scripts and the data directory

**Examples**

```
## Not run:  
rdataretriever::reset("iris")  
  
## End(Not run)
```

---

`socrata_autocomplete_search`*Returns the list of dataset names after autocompletion*

---

**Description**

Returns the list of dataset names after autocompletion

**Usage**

```
socrata_autocomplete_search(dataset)
```

**Arguments**

`dataset`            the name of the dataset

**Value**

No return value, show dataset names after autocompletion

**Examples**

```
## Not run:  
rdataretriever::socrata_autocomplete_search()  
  
## End(Not run)
```

---

`socrata_dataset_info`    *Get socrata dataset info*

---

**Description**

Get socrata dataset info

**Usage**

```
socrata_dataset_info(dataset_name)
```

**Arguments**

`dataset_name`    dataset name to obtain info

**Value**

No return value, shows socrata dataset info

**Examples**

```
## Not run:  
rdataretriever::socrata_dataset_info()  
  
## End(Not run)
```

---

update\_rdataset\_catalog

*Updates the datasets\_url.json from the github repo*

---

**Description**

Updates the datasets\_url.json from the github repo

**Usage**

```
update_rdataset_catalog(test = FALSE)
```

**Arguments**

test                    flag set when testing

**Value**

No return value, updates the datasets\_url.json

**Examples**

```
## Not run:  
rdataretriever::update_rdataset_catalog()  
  
## End(Not run)
```

---

use\_RetrieverPath

*Setting path of retriever*

---

**Description**

Setting path of retriever

**Usage**

```
use_RetrieverPath(path)
```

**Arguments**

path                    location of retriever in the system



**Value**

No return value, setting path of retriever

**Examples**

```
## Not run:  
rdataretriever::use_RetrieverPath("/home/<system_name>/anaconda2/envs/py27/bin/")  
  
## End(Not run)
```

# Index

## \* **utilities**

- get\_updates, [12](#)
- check\_for\_updates, [2](#)
- check\_retriever\_availability, [3](#)
- commit, [4](#)
- commit\_log, [4](#)
- conda\_binary(), [19](#)
- data\_retriever\_version, [6](#)
- dataset\_names, [6](#)
- datasets, [5](#)
- display\_all\_rdataset\_names, [7](#)
- download, [7](#)
- fetch, [8](#)
- find\_socrata\_dataset\_by\_id, [9](#)
- get\_dataset\_names\_upstream, [10](#)
- get\_rdataset\_names, [10](#)
- get\_retriever\_citation, [11](#)
- get\_script\_citation, [11](#)
- get\_script\_upstream, [12](#)
- get\_updates, [12](#)
- install, [13](#)
- install\_csv, [14](#)
- install\_json, [15](#)
- install\_msaccess, [16](#)
- install\_mysql, [17](#)
- install\_postgres, [18](#)
- install\_retriever, [19](#)
- install\_sqlite, [20](#)
- install\_xml, [21](#)
- reload\_scripts, [22](#)
- reset, [22](#)
- socrata\_autocomplete\_search, [23](#)
- socrata\_dataset\_info, [23](#)
- update\_rdataset\_catalog, [24](#)
- use\_RetrieverPath, [24](#)