Package 'ohvbd'

March 9, 2026

Title One Health VBD Hub

Version 1.0.1

Description Interface with the One Health VBD (vector-borne disease) Hub <https://vbdhub.org/> and related repositories (VectorByte <https://www.vectorbyte.org>, GBIF <https://www.gbif.org> and AREAdata <https://pearselab.github.io/areadata/>) directly to find, download, and subset vector-borne disease data.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/fwimp/ohvbd, https://ohvbd.vbdhub.org

BugReports https://github.com/fwimp/ohvbd/issues

Imports cli (>= 3.0.0), curl (>= 6.4.0), data.table, dplyr, generics, httr2 (>= 1.1.1), lubridate, rlang (>= 1.1.0), stats, stringr, terra, tibble, tidyr, utils

Suggests knitr, parsermd, rgbif, rmarkdown, rnaturalearth, testthat (>= 3.0.0), vcr, withr

VignetteBuilder knitr

Depends R (>= 4.1.0)

Config/testthat/edition 3

Config/testthat/parallel true

NeedsCompilation no

Author Francis Windram [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-2129-826X>)

Maintainer Francis Windram <francis.windram17@imperial.ac.uk>

Repository CRAN

Date/Publication 2026-03-09 17:20:02 UTC

Contents

`ad_basereq` *Generate a base request string for the AREAdata database*

Description

This string is used as the basis for all calls to AREAdata. It does not contain any tokens or session ids, and thus can be regenerated at any time.

Usage

```
ad_basereq()
```

Value

Returns a string containing the root address of the AREAdata dataset.

Author(s)

Francis Windram

Examples

```
basereq <- ad_basereq()
```

`assoc_ad` *Associate other data sources with AREAdata data*

Description

Intelligently bind together data from AREAdata and other sources at a given spatial scale.

Usage

```
assoc_ad(
  data,
  areadata,
  targetdate = NA,
  enddate = NA,
  gid = 0,
  lonlat_names = c("Longitude", "Latitude"),
  cache_location = NULL,
  basereq = ad_basereq()
)
```

Arguments

`data`	the source data to bind AREAdata to. This **must** contain decimal lonlat data!
`areadata`	the AREAdata to bind, usually from `fetch_ad()`.
`targetdate`	**ONE OF** the following: • The date to search for in ISO 8601 (e.g. "2020", "2021-09", or "2022-09-21"). • The start date for a range of dates. • A character vector of fully specified dates to search for (i.e. "yyyy-mm-dd").
`enddate`	The (exclusive) end of the range of dates to search for. If this is unfilled, only the `targetdate` is searched for.
`gid`	the spatial scale to retrieve (0 = country-level, 1=province-level...). (Note: this will preferentially use the gid level of `areadata` if present.)
`lonlat_names`	a vector containing the column names of the longitude and latitude columns **IN THAT ORDER**!
`cache_location`	path to cache location (defaults to a temporary user directory, or one set by `set_default_ohvbd_cache()`).
`basereq`	the url of the AREAdata database (usually generated by `ad_basereq()`). If `NA`, uses the default.

Value

A matrix of the `data` with added columns extracted from `areadata`.

Date ranges

The date range is a partially open interval. That is to say the lower bound (`targetdate`) is inclusive, but the upper bound (`enddate`) is exclusive.

For example a date range of "2020-08-04" - "2020-08-12" will return the 7 days from the 4th through to the 11th of August, but *not* the 12th.

Date inference

In cases where a full date is not provided, the earliest date possible with the available data is chosen.

So "2020-04" will internally become "2020-04-01".

If an incomplete date is specified as the `targetdate` and no `enddate` is specified, the range to search is inferred from the minimum temporal scale provided in `targetdate`.

For example "2020-04" will be taken to mean the month of April in 2020, and the `enddate` will internally be set to "2020-05-01".

Author(s)

Francis Windram

Examples

```
vtdf <- search_hub("Aedes aegypti", "vt") |>
  tail(20) |>
  fetch() |>
  glean(cols = c(
    "DatasetID",
    "Latitude",
    "Longitude",
    "Interactor1Genus",
    "Interactor1Species"
    ), returnunique = TRUE)
areadata <- fetch_ad(metric="temp", gid=2, use_cache=TRUE)
ad_extract_working <- assoc_ad(vtdf, areadata,
                                   targetdate = c("2021-08-04"), enddate=c("2021-08-06"),
                                     gid=2, lonlat_names = c("Longitude", "Latitude"))
```

`assoc_gadm` *Associate other data sources with gadm IDs*

Description

Intelligently bind together data with gadm IDs at all scales.

Usage

```
assoc_gadm(
  df,
  lonlat_names = c("Longitude", "Latitude"),
  cache_location = NULL,
  basereq = ad_basereq()
)
```

Arguments

df	the source data to bind gadm IDs to. This **must** contain decimal lonlat data!
lonlat_names	a vector containing the column names of the longitude and latitude columns **IN THAT ORDER**!
cache_location	path to cache location (defaults to a temporary user directory, or one set by set_default_ohvbd_cache()).
basereq	the url of the AREAdata database (usually generated by ad_basereq()). If NA, uses the default.

Value

A matrix of the data with added gadm columns.

Caching

This will **ALWAYS** get and cache gid level 2 data sources. These files are about 80MB total, so if you are running on a metered connection do beware of this.

Author(s)

Francis Windram

Examples

```
vtdf <- search_hub("Aedes aegypti", "vt") |>
  tail(20) |>
  fetch() |>
  glean(cols = c(
    "DatasetID",
    "Latitude",
    "Longitude",
    "Interactor1Genus",
    "Interactor1Species"
    ), returnunique = TRUE) |>
  assoc_gadm(lonlat_names = c("Longitude", "Latitude"))
```

---

check_db_status *Check whether databases are currently online*

Description

Attempt to access all presently supported databases and report if they were accessible.

Usage

```
check_db_status()
```

Value

TRUE if all DB checks pass, else FALSE

Author(s)

Francis Windram

Examples

```
check_db_status()
```

`check_ohvbd_config` *Print current ohvbd configuration variables*

Description

Access ohvbd options and configured variables, and print them to the command line.

Usage

```
check_ohvbd_config(options_list = NULL)
```

Arguments

`options_list`	An (optional) list of variables to search for.

Value

`TRUE` if all desired options are set (though not necessarily turned on), else `FALSE`.

Author(s)

Francis Windram

Examples

```
check_ohvbd_config()
```

clean_ad_cache *Delete all rda files from ohvbd AREAdata cache (Deprecated)*

Description

Delete all rda files from ohvbd AREAdata cache (Deprecated)

Usage

```
clean_ad_cache(cache_location = NULL)
```

Arguments

cache_location	location of the cache.

Value

No return value, called for side effects

Note

`clean_ad_cache()` is now deprecated and should not be used. Please use `clean_ohvbd_cache()` instead.

Author(s)

Francis Windram

clean_ohvbd_cache *Delete files from ohvbd cache directories*

Description

Delete files from ohvbd cache directories

Usage

```
clean_ohvbd_cache(subdir = NULL, path = NULL, dryrun = FALSE, force = FALSE)
```

Arguments

subdir	a subdirectory or list of subdirectories to clean.
path	location within which to remove rda files. (Defaults to the standard ohvbd cache location).
dryrun	if TRUE list files that would be deleted, but do not remove.
force	do not ask for confirmation before cleaning.

Value

No return value, called for side effects

Author(s)

Francis Windram

Examples

```
clean_ad_cache()
```

`extract` *Extract specified data from a set of responses (Deprecated)*

Description

This is a convenience method that infers and applies the correct extractor for the input.

Usage

```
extract(res, ...)
```

Arguments

`res`	An object of type `ohvbd.responses` or `ohvbd.ad.matrix` generated from `fetch()` and containing data from one of the supported databases.
`...`	Any arguments to be passed to the underlying extractors (see `glean_vt()` and `glean_ad()` for specific arguments).

Value

The extracted data, either as an `ohvbd.data.frame` or `ohvbd.ad.matrix` object.

Note

`extract()` is now deprecated and should not be used. Please use `glean()` instead.

Author(s)

Francis Windram

fetch *Fetch specified data from a set of ids*

Description

This is a convenience method that infers and applies the correct fetch function for the input ids.

Usage

```
fetch(ids, ...)
```

Arguments

ids	An object of type ohvbd.ids (generated from a search, manually packaged using ohvbd.ids() or generated by another function).
...	Any other arguments to be passed to the underlying fetch functions (see fetch_vt() and fetch_vd() for specific arguments).

Value

The downloaded data, as an ohvbd.responses object.

Author(s)

Francis Windram

Examples

```
search_hub("Ixodes", "vt") |> fetch()
```

fetch_ad *Fetch AREAdata dataset*

Description

Retrieve AREAdata dataset/s specified by metric and spatial scale (GID).

Usage

```
fetch_ad(
  metric = "temp",
  gid = 0,
  use_cache = TRUE,
  cache_location = NULL,
  refresh_cache = FALSE,
  timeout = 240,
  basereq = ad_basereq()
)
```

Arguments

metric	the metric to retrieve from areadata.
gid	the spatial scale to retrieve (0 = country-level, 1=province-level ...).
use_cache	load files from cache if possible, and save them if not present.
cache_location	path to cache location (defaults to a temporary user directory, or one set by set_default_ohvbd_cache()).
refresh_cache	force a refresh of the relevant cached data (and enables use_cache).
timeout	timeout for data download from figshare/github in seconds.
basereq	the url of the AREAdata database (usually generated by ad_basereq()). If NA, uses the default.

Value

A ohvbd.ad.matrix of the requested data (with added attributes for gid and metric).

Valid metrics

The following metrics are valid (alternative names are listed in brackets):

- temp (*temperature*)
- spechumid (*specific humidity*)
- relhumid (*relative humidity*)
- uv (*ultraviolet*)
- precip (*precipitation, rainfall*)
- popdens (*population density, population*)
- forecast (*future climate, future*)

Author(s)

Francis Windram

Examples

```
fetch_ad(metric="temp", gid=0)
```

fetch_citations *Try to find the relevant citations for a dataset*

Description

This tries to extract and simplify the citations from a dataset downloaded using ohvbd.

Usage

```
fetch_citations(dataset, ...)
```

Arguments

dataset	An object of type ohvbd.data.frame (generated from glean(), preferred) or of type ohvbd.ids and containing data from one of the supported databases.
...	Any arguments to be passed to the underlying funcs.

Value

The extracted data, either as an ohvbd.data.frame or ohvbd.ad.matrix object.

Author(s)

Francis Windram

Examples

```
search_hub("Ixodes", "vt") |>
  fetch() |>
  glean() |>
  fetch_citations()
```

fetch_citations_vd *Retrieve citations for vecdyn data*

Description

Retrieve citations for vecdyn data either directly from the dataset or by redownloading the appropriate data.

Usage

```
fetch_citations_vd(
  dataset,
  redownload = TRUE,
  minimise = FALSE,
  collapse_cols = TRUE
)
```

Arguments

dataset	The dataset from which you wish to retrieve citations.
redownload	Redownload data if citation columns are missing.
minimise	Whether to return one row per citation (rather than one per dataset ID).
collapse_cols	Whether to remove completely empty columns.

Value

ohvbd.data.frame of citation data

Author(s)

Francis Windram

---

fetch_citations_vt *Retrieve citations for vectraits data*

Description

Retrieve citations for vectraits data either directly from the dataset or by redownloading the appropriate data.

Usage

```
fetch_citations_vt(dataset, redownload = TRUE, minimise = FALSE)
```

Arguments

dataset	The dataset from which you wish to retrieve citations.
redownload	Redownload data if citation columns are missing.
minimise	Whether to return one row per citation (rather than one per dataset ID).

Value

ohvbd.data.frame of citation data

Author(s)

Francis Windram

fetch_gadm_sfs *Fetch gadm mapping shapefiles*

Description

Retrieve AREAdata gadm mapping shapefiles specified by spatial scale (GID). These vectors are cached as GeoPackage files.

Usage

```
fetch_gadm_sfs(
  gid = 0,
  cache_location = NULL,
  refresh_cache = FALSE,
  basereq = ad_basereq(),
  call = rlang::caller_env()
)
```

Arguments

gid	the spatial scale to retrieve (0 = country-level, 1=province-level...).
cache_location	path to cache location (defaults to a temporary user directory, or one set by set_default_ohvbd_cache()).
refresh_cache	force a refresh of the relevant cached data.
basereq	the url of the AREAdata database (usually generated by ad_basereq()). If NA, uses the default.
call	The env from which this was called (defaults to the direct calling environment).

Value

A SpatVector (from terra::vect()) of the requested shapefile.

Author(s)

Francis Windram

Examples

```
fetch_gadm_sfs(gid=0)
```

`fetch_gbif` *Fetch GBIF dataset/s by ID*

Description

Retrieve GBIF dataset/s specified by their dataset ID.

Usage

```
fetch_gbif(ids, filepath = ".")
```

Arguments

`ids`	a string or character vector of ids (preferably in an `ohvbd.ids` object) indicating the particular dataset/s to download.
`filepath`	directory to save gbif download files into.

Value

A list of rgbif occ_download_get objects, as an `ohvbd.responses` object.

Note

Only 300 datasets can be requested at once (for now) due to technical limitations originating from the GBIF server setup. It is worth splitting longer lists of ids into a couple of chunks if you need more than this.

If you regularly use ohvbd to download large numbers of datasets at once and chunking is causing you other issues, please raise an issue at https://github.com/fwimp/ohvbd/issues.

Author(s)

Francis Windram

Examples

```
fetch_gbif("dbc4a3ae-680f-44e6-ab25-c70e27b38dbc")

ohvbd.ids("dbc4a3ae-680f-44e6-ab25-c70e27b38dbc", "gbif") |>
  fetch() # Calls fetch_gbif()
```

`fetch_glean_vd_chunked` *Fetch and parse multiple VecDyn datasets by ID in chunks*

Description

Retrieve and parse VecDyn datasets specified by their dataset IDs in batches.

This is not usually necessary (generally you just need `fetch_vd()`) but allows one to release data that is not in use from memory. If you would like more control on extraction or parsing then it is best to wrap `fetch_vd()` and `glean_vd()` in your own chunker instead.

Usage

```
fetch_glean_vd_chunked(
  ids,
  chunksize = 20,
  cols = NULL,
  returnunique = FALSE,
  rate = 5,
  connections = 2,
  basereq = vb_basereq()
)
```

Arguments

`ids`	a numeric vector of IDs (preferably in an `ohvbd.ids` object) indicating the particular datasets to download.
`chunksize`	an integer defining the size of chunks to retrieve in one iteration.
`cols`	a character vector of columns to extract from the dataset.
`returnunique`	whether to return only the unique rows within each dataset according to the filtered columns.
`rate`	maximum number of calls to the API per second.
`connections`	number of simultaneous connections to the server at once. Maximum 8. **Do not enable unless you really need to** as this hits the server significantly harder than usual.
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

An `ohvbd.data.frame` containing the requested data.

Author(s)

Francis Windram

Examples

```
fetch_glean_vd_chunked(c(423,424,425), chunksize = 2, rate=5)
```

`fetch_glean_vt_chunked` *Get and parse multiple VecTraits datasets by ID in chunks*

Description

Retrieve and parse VecTraits datasets specified by their dataset IDs in batches.

This is not usually necessary (generally you just need `fetch_vt()`) but allows one to release data that is not in use from memory. If you would like more control on extraction or parsing then it is best to wrap `fetch_vt()` and `glean_vt()` in your own chunker instead.

Usage

```
fetch_glean_vt_chunked(
  ids,
  chunksize = 20,
  cols = NULL,
  returnunique = FALSE,
  rate = 5,
  connections = 2,
  basereq = vb_basereq()
)
```

Arguments

`ids`	a numeric vector of IDs (preferably in an `ohvbd.ids` object) indicating the particular datasets to download.
`chunksize`	an integer defining the size of chunks to retrieve in one iteration.
`cols`	a character vector of columns to extract from the dataset.
`returnunique`	whether to return only the unique rows within each dataset according to the filtered columns.
`rate`	maximum number of calls to the API per second.
`connections`	number of simultaneous connections to the server at once. Maximum 8. **Do not enable unless you really need to** as this hits the server significantly harder than usual.
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

An `ohvbd.data.frame` containing the requested data.

Author(s)

Francis Windram

Examples

```
fetch_glean_vt_chunked(c(54,55,56), chunksize = 2, rate=5)
```

---

fetch_vd *Fetch VecDyn dataset/s by ID*

Description

Retrieve VecDyn dataset/s specified by their dataset ID.

Usage

```
fetch_vd(ids, rate = 5, connections = 2, basereq = vb_basereq())
```

Arguments

ids	a numeric ID or numeric vector of ids (preferably in an ohvbd.ids object) indicating the particular dataset/s to download.
rate	maximum number of calls to the API per second.
connections	number of simultaneous connections to the server at once. Maximum 8. **Do not enable unless you really need to** as this hits the server significantly harder than usual.
basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

A list of httr2 response objects, as an ohvbd.responses object.

Author(s)

Francis Windram

Examples

```
fetch_vd(54)

fetch_vd(c(423,424,425))

ohvbd.ids(c(423,424,425), "vd") |>
  fetch() # Calls fetch_vd()
```

`fetch_vd_counts` *Fetch VecDyn dataset length by ID*

Description

Retrieve length of VecDyn dataset/s specified by their dataset ID.

Usage

```
fetch_vd_counts(
  ids,
  page_size = 50,
  cache_location = NULL,
  refresh_cache = FALSE,
  noprogress = FALSE,
  basereq = vb_basereq()
)
```

Arguments

`ids`	a numeric ID or numeric vector of ids (preferably in an `ohvbd.ids` object) indicating the particular dataset/s to download.
`page_size`	the page size returned by VecDyn (default is 50).
`cache_location`	path to cache location (defaults to a temporary user directory, or one set by `set_default_ohvbd_cache()`).
`refresh_cache`	force a refresh of the relevant cached data.
`noprogress`	disable non-essential messaging (progress bars etc.).
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

A dataframe describing the number of rows and number of pages for the set of ids.

Author(s)

Francis Windram

Examples

```
fetch_vd_counts(54)

fetch_vd_counts(c(423,424,425))
```

`fetch_vt` *Fetch VecTraits dataset/s by ID*

Description

Retrieve VecTraits dataset/s specified by their dataset ID.

Usage

```
fetch_vt(ids, rate = 5, connections = 2, basereq = vb_basereq())
```

Arguments

`ids`	a numeric ID or numeric vector of ids (preferably in an `ohvbd.ids` object) indicating the particular dataset/s to download.
`rate`	maximum number of calls to the API per second.
`connections`	number of simultaneous connections to the server at once. Maximum 8. **Do not enable unless you really need to** as this hits the server significantly harder than usual.
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

A list of httr2 response objects, as an `ohvbd.responses` object.

Author(s)

Francis Windram

Examples

```
fetch_vt(54)

fetch_vt(c(54, 55, 56))

ohvbd.ids(c(54, 55, 56), "vt") |>
  fetch() # Calls fetch_vt()
```

`filter_db` *Filter hub search results by database*

Description

Retrieve the IDs for any datasets matching the given database.

Usage

```
filter_db(ids, db)
```

Arguments

`ids`	an `ohvbd.hub.search` search result from `search_hub()`.
`db`	a database name as a string. One of `"vt"`, `"vd"`, `"gbif"`, `"px"`.

Value

An `ohvbd.ids` vector of dataset IDs.

Note

If `filter_db()` recieves an `ohvbd.ids` object by mistake, it will transparently return it if the source database matches `db`.

Author(s)

Francis Windram

Examples

```
search_hub("Ixodes ricinus")

search_hub("Ixodes ricinus") |>
  filter_db("vt") |>
  fetch() |>
  glean()
```

find_vd_columns *Get current columns in VecDyn datasets*

Description

Get all the current column names in VecDyn, alongside associated data if desired.

Usage

```
find_vd_columns(full = FALSE, basereq = vb_basereq())
```

Arguments

full	whether to return all the data about current fields, or else just return the names
basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

A character vector (or dataframe) of column information.

Author(s)

Francis Windram

Examples

```
find_vd_columns()
```

find_vd_ids *Get current IDs in VecDyn*

Description

Get all the current dataset IDs in VecDyn, as a numeric vector.

Usage

```
find_vd_ids(basereq = vb_basereq())
```

Arguments

basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

An ohvbd.ids vector of VecDyn dataset IDs.

Author(s)

Francis Windram

Examples

```
find_vd_ids()
```

`find_vt_ids` *Get current IDs in VecTraits*

Description

Get all the current dataset IDs in VecTraits, as a numeric vector.

Usage

```
find_vt_ids(basereq = vb_basereq())
```

Arguments

basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

An ohvbd.ids vector of VecTraits dataset IDs.

Author(s)

Francis Windram

Examples

```
find_vt_ids()
```

`force_db` *Force an object to appear to come from a specific database*

Description

Force an object to appear to come from a specific database

Usage

```
force_db(x, db)
```

Arguments

`x`	Object to force.
`db`	Database to apply to `x`.

Value

Object with the "db" attribute set to `db`

Note

DO NOT use this function to create ids to feed into `fetch()`!

Objects created in this way may lack vital underlying data required later. Instead use `ohvbd.ids()` for this purpose.

This is a synonym for `ohvbd_db(x) <- db` that's easier to work with in pipelines.

Author(s)

Francis Windram

See Also

Internal attributes

Examples

```
force_db(c(1,2,3), "vt")
```

`format_time_overlap_bar` *Format and print date overlaps*

Description

Format and output to the terminal a visualisation of the overlaps between a given period and another set of dates. This is mostly used in the error handling of `glean_ad()` however it can also be used independently. It was designed to fill a more general role within UI design using the cli package, and should be usable (or hackable) by others needing the same tool.

Usage

```
format_time_overlap_bar(
  start,
  end,
  targets,
  targetrange = FALSE,
  twobar = FALSE,
  width = NULL,
  style = list()
)
```

Arguments

`start`	the date that the reference period begins (as Date object).
`end`	the date that the reference period ends (as Date object).
`targets`	a vector of dates.
`targetrange`	is the target a range? If so this will treat the first two elements of `targets` as the start and end of the range.
`twobar`	whether to render as two bars or as one with different colours for overlaps.
`width`	the width of the bars in characters. Defaults to 0.5 * console width.
`style`	a style from `cli::cli_progress_styles()` to use as a format.

Value

No return value

Author(s)

Francis Windram

Examples

```
format_time_overlap_bar(
  start = as.Date("2022-08-04"),
  end = as.Date("2022-08-11"),
  targets = c(as.Date("2022-08-05"), as.Date("2022-08-12")),
  targetrange = TRUE, twobar=TRUE
)
```

`generate_vt_template` *Generate a vectraits template from a short set of survey responses*

Description

Given the large number of fields in vectraits it can be hard to know which of these you need to fill out. This generator asks a series of questions to determine what columns should be included in one's dataset.

Usage

```
generate_vt_template()
```

Value

A character vector containing the column headers of the desired vectraits template or NULL.

Author(s)

Francis Windram

Examples

```
generate_vt_template()
```

get_default_ohvbd_cache *Get ohvbd cache locations*

Description

Get ohvbd cache locations

Usage

```
get_default_ohvbd_cache(subdir = NULL, create = TRUE)
```

Arguments

subdir	The subdirectory within the cache to find/create (optional).
create	Whether to create the cache location if it does not already exist (defaults to TRUE).

Value

ohvbd cache path as a string

Author(s)

Francis Windram

Examples

```
get_default_ohvbd_cache()
```

glean *Extract specified data from a set of responses*

Description

This is a convenience method that infers and applies the correct extractor for the input

Usage

```
glean(res, ...)
```

Arguments

res	An object of type ohvbd.responses or ohvbd.ad.matrix generated from fetch() and containing data from one of the supported databases.
...	Any arguments to be passed to the underlying extractors (unused).

Value

The extracted data, either as an `ohvbd.data.frame` or `ohvbd.ad.matrix` object.

Author(s)

Francis Windram

Examples

```
search_hub("Ixodes", "vt") |> fetch() |> glean(cols=c("Interactor1Species"))
fetch_ad(use_cache=TRUE) |> glean(targetdate="2020-08-04")
```

`glean_ad` *Extract data from AREAdata datasets*

Description

Extract the data returned by a call to `fetch_ad()`, filter columns of interest and by dates of interest.

Currently this does not handle Population Density or Forecast matrices, however the other 5 metrics are handled natively.

Usage

```
glean_ad(
  ad_matrix,
  targetdate = NA,
  enddate = NA,
  places = NULL,
  gid = NULL,
  printbars = TRUE
)
```

Arguments

`ad_matrix`	A matrix or `ohvbd.ad.matrix` of data from AREAdata.
`targetdate`	**ONE OF** the following: • The date to search for in ISO 8601 (e.g. "2020", "2021-09", or "2022-09-21"). • The start date for a range of dates. • A character vector of fully specified dates to search for (i.e. "yyyy-mm-dd")
`enddate`	The (exclusive) end of the range of dates to search for. If this is unfilled, only the `targetdate` is searched for.
`places`	A character vector or single string describing what locality to search for in the dataset.

gid	The spatial scale of the AREAdata matrix (this is not needed if the matrix has been supplied by fetch_ad()).
printbars	Whether to print time overlap bars in the case of dates outside the data range.

Value

An ohvbd.ad.matrix or a named vector containing the extracted data.

Place matching

This function attempts to intelligently infer place selections based upon the provided gid and place names.

So if you have an AREAdata dataset at gid=1, and provide country names, the function will attempt to match those country names and retrieve any GID1-level data that is present.

Occasionally (such as in the case of "Albania", the municipality in La Guajira, Columbia) the name of a place may occur in locations other than those expected by the researcher.

Unfortunately this is not an easy problem to mitigate, and as such it is worthwhile checking the output of this function to make sure it is as you expect.

Date ranges

The date range is a partially open interval. That is to say the lower bound (targetdate) is inclusive, but the upper bound (enddate) is exclusive.

For example a date range of "2020-08-04" - "2020-08-12" will return the 7 days from the 4th through to the 11th of August, but *not* the 12th.

Date inference

In cases where a full date is not provided, the earliest date possible with the available data is chosen.

So "2020-04" will internally become "2020-04-01".

If an incomplete date is specified as the targetdate and no enddate is specified, the range to search is inferred from the minimum temporal scale provided in targetdate.

For example "2020-04" will be taken to mean the month of April in 2020, and the enddate will internally be set to "2020-05-01".

Author(s)

Francis Windram

Examples

```
# All dates in August 2022
fetch_ad("temp", gid=0) |>
  glean_ad(
    targetdate = "2022-08",
    places = c("Albania", "Thailand")
  )
```

```
# 4th, 5th, and 6th of August 2022 (remember the enddate is EXCLUSIVE)
fetch_ad("temp", gid=0) |>
  glean_ad(
    targetdate = "2022-08-04", enddate="2022-08-07",
    places = c("Albania", "Thailand")
  )

# 4th of August 2022 and 1st of August 2023
fetch_ad("temp", gid=0) |>
  glean_ad(
    targetdate = c("2022-08-04", "2023-08-01"),
    places = c("Albania", "Thailand")
  )
```

`glean_gbif` *Parse data from requests to GBIF*

Description

Extract the data returned by a call to `fetch_gbif()`, filter columns of interest, and find unique rows if required.

Usage

```
glean_gbif(res, cols = NULL, returnunique = FALSE)
```

Arguments

`res`	a list of responses from GBIF as an `ohvbd.responses` object.
`cols`	a character vector of columns to extract from the dataset.
`returnunique`	whether to return only the unique rows within each dataset according to the filtered columns.

Value

An `ohvbd.data.frame` containing the requested data.

Author(s)

Francis Windram

Examples

```
fetch_gbif("dbc4a3ae-680f-44e6-ab25-c70e27b38dbc") |>
  glean_gbif()

ohvbd.ids("dbc4a3ae-680f-44e6-ab25-c70e27b38dbc", "gbif") |>
  fetch() |>
  glean() # Calls glean_gbif()
```

---

glean_vd *Parse data from requests to VecDyn*

Description

Extract the data returned by a call to `fetch_vd()`, filter columns of interest, and find unique rows if required.

Usage

```
glean_vd(res, cols = NULL, returnunique = FALSE)
```

Arguments

`res`	a list of responses from VecDyn as an `ohvbd.responses` object.
`cols`	a character vector of columns to extract from the dataset. If specified, this will be adjusted to always include the "dataset_id" column.
`returnunique`	whether to return only the unique rows within each dataset according to the filtered columns.

Value

An `ohvbd.data.frame` containing the requested data.

Author(s)

Francis Windram

Examples

```
fetch_vd(247) |>
  glean_vd(cols=c("species",
                  "sample_start_date",
                  "sample_value"),
           returnunique=TRUE)

ohvbd.ids(247, "vd") |>
  fetch() |>
  glean() # Calls glean_vd()
```

`glean_vt` *Parse data from requests to VecTraits*

Description

Extract the data returned by a call to `fetch_vt()`, filter columns of interest, and find unique rows if required.

Usage

```
glean_vt(res, cols = NULL, returnunique = FALSE)
```

Arguments

`res`	a list of responses from VecTraits as an `ohvbd.responses` object.
`cols`	a character vector of columns to extract from the dataset.
`returnunique`	whether to return only the unique rows within each dataset according to the filtered columns.

Value

An `ohvbd.data.frame` containing the requested data.

Author(s)

Francis Windram

Examples

```
fetch_vt(54) |>
  glean_vt(cols=c("DatasetID",
                  "Interactor1Genus",
                  "Interactor1Species"),
           returnunique=TRUE)

ohvbd.ids(54, "vt") |>
  fetch() |>
  glean() # Calls glean_vt()
```

has_db *Test whether an object has provenance information*

Description

This function tests whether an object has the provenance information expected by ohvbd.

Usage

```
has_db(x, ...)
```

Arguments

x	An object to test.
...	Any arguments to be passed to the underlying functions (unused).

Value

Whether the data has a provenance tag (as a boolean).

Author(s)

Francis Windram

Examples

```
ids <- ohvbd.ids(c(1,2,3), "vd")
has_db(ids)
```

is_cached *Check whether an object has been loaded from cache by ohvbd*

Description

Check whether an object has been loaded from cache by ohvbd

Usage

```
is_cached(x)
```

Arguments

x	The object to check.

Value

A boolean indicating whether an object has been loaded from the cache.

Author(s)

Francis Windram

Examples

```
is.cached(c(1,2,3))
```

`is_from` *Test whether an object is considered to be from a particular database*

Description

This function tests whether an object is considered (by ohvbd) to be from a given database.

This is a fairly coarse check, and so cannot "work out" data provenance from its structure.

Usage

```
is_from(x, db, ...)
```

Arguments

x	An object to test.
db	The database to test against.
...	Any arguments to be passed to the underlying functions (unused).

Value

Whether the data is from a given database (as a boolean).

Author(s)

Francis Windram

Examples

```
ids <- ohvbd.ids(c(1,2,3), "vd")
is_from(ids, "vd")
```

list_ohvbd_cache *List all ohvbd cached files*

Description

List all ohvbd cached files

Usage

```
list_ohvbd_cache(subdir = NULL, path = NULL, treeview = FALSE)
```

Arguments

subdir	a subdirectory or list of subdirectories to list.
path	location within which to list files. (Defaults to the standard ohvbd cache location).
treeview	display the full cache in a tree structure

Value

No return value

Author(s)

Francis Windram

Examples

```
list_ohvbd_cache()
```

match_countries *Match country names to their equivalent naturalearth WKT polygons*

Description

Match country names to their equivalent naturalearth WKT polygons using `rnaturalearth::ne_countries()`.

Usage

```
match_countries(countrynames, returnmulti = TRUE, onlywkt = FALSE)
```

Arguments

countrynames	a vector of country names to match to naturalearth.
returnmulti	return the GBIF taxon ids only (otherwise return the full lookup dataframe).
onlywkt	only return location_wkt (see note for more details).

Value

A list containing:

- `$location_wkt`: a multipolygon containing all locations (or a named vector of individual country polygons).
- `$missing_locs`: any provided countries not found in naturalearth.
- `$found_locs`: any provided countries that were found in naturalearth.

Author(s)

Francis Windram

Examples

```
match_countries(c("United Kingdom", "Germany"))
```

---

match_species *Match species names to their GBIF backbone ids*

Description

Match species names to their GBIF backbone ids using `rgbif::name_backbone_checklist()`.

Usage

```
match_species(speciesnames, exact = FALSE, returnids = TRUE, omit = TRUE)
```

Arguments

`speciesnames`	a vector of species names to match to the GBIF backbone.
`exact`	whether to only return exact *species* matches.
`returnids`	return the GBIF taxon ids only (otherwise return the full lookup dataframe).
`omit`	omit missing taxon ids (inactive when `returnids = FALSE`).

Value

The GBIF taxonids associated with `speciesnames` or the full GBIF lookup dataframe if `returnids = TRUE`.

Note

If `exact = TRUE` and you search for a genus name, this will not be returned. If you want more control over id filtering, use `returnids = FALSE` to get the source dataframe.

Author(s)

Francis Windram

Examples

```
match_species(c("Araneus diadematus", "Aedes aegypti"))
```

`ohvbd.ids` *Create a new ohvbd ID vector*

Description

When retrieving data from previous searches (or saved lists of IDs), it can be useful to package these data up in the form that ohvbd would expect to come out of a search.

To do this, create an `ohvbd.ids` object, specifying the database that the ids refer to.

Usage

```
ohvbd.ids(ids, db)
```

Arguments

`ids`	A numeric vector of ids referring to datasets within the specified database.
`db`	A string specifying the database that these ids refer to.

Value

An id vector: an S3 vector with class `ohvbd.ids`.

Author(s)

Francis Windram

Examples

```
ohvbd.ids(c(1,2,3,4,5), db="vt")

ohvbd.ids(c(1,2,3,4,5), db="vd")

ohvbd.ids(
  c(
    "dbc4a3ae-680f-44e6-ab25-c70e27b38dbc",
    "fac87892-68c8-444a-9ae9-46273fdff724"
    ),
  db="gbif"
)
```

ohvbd_attrs *Internal attributes*

Description

ohvbd uses a number of internal attributes to track data states within pipelines. Generally these are not designed to be user-modified. They are, however, listed here for completeness (and curiosity).

It is typically not a good idea to manually modify these attributes directly unless a helper such as `force_db()` or `ohvbd_db()` is provided. Even then, modifying these attributes may cause unexpected errors or data inconsistencies. These errors may not be signalled to the user by ohvbd, and they may not be obvious *or even detectable*.

Be sure when modifying the db attribute that the value you set it to is consistent with the origin of your data, and that the value is a db known to ohvbd.

Summary table of attributes

Attribute	Description	Object/s
db	The database from which the object has been retrieved.	`ohvbd.ids`, `ohvbd.responses`, `ohvbd.data.f`
metric	The AD metric.	`ohvbd.ad.matrix`
gid	The AD aggregation level.	`ohvbd.ad.matrix`
cached	Whether the data was loaded from a cache.	Any
writetime	The time at which a data file was originally cached.	Any
query	The search query sent to the Hub.	`ohvbd.hub.search`
searchparams	Any extra parameters sent to the Hub.	`ohvbd.hub.search`

Note: (AD = AREAdata)

db

Type: string

The **db** attribute indicates to ohvbd where an object originated. It is used to determine appropriate method dispatch (such as with `fetch()`) and to check that pipelines are sensible constructed.

metric

Type: string

metric signifies what AD metric the matrix contains. It is predominantly (but not exclusively) used for formatting and caching.

gid

Type: integer

gid represents the spatial scale of data from AD. It is used for a variety of spatial operations.

cached

Type: boolean

cached objects receive this flag at write-time. It sticks with the object when it is reloaded, and is mostly used for UI/UX purposes.

writetime

Type: POSIXct

writetime stores the time at which a cached object (that is likely to become stale) was written to the cache.

query

Type: string

Simply stores the base **query** that was sent to the vbdhub search API.

searchparams

Type: named list

A record of any other search parameters that were sent to the vbdhub search API (e.g. species IDs etc.).

Author(s)

Francis Windram

`ohvbd_db` *Database provenance*

Description

Retrieve or set the provenance information expected by `ohvbd`.

Usage

```
ohvbd_db(x)

ohvbd_db(x) <- value
```

Arguments

`x`	An object.
`value`	The value to set the db to.

Value

The database identifier associated with an object (or `NULL` if missing).

Author(s)

Francis Windram

See Also

Internal attributes

Examples

```
ids <- ohvbd.ids(c(1,2,3), "vd")
ohvbd_db(ids)

ohvbd_db(ids) <- "vt"
ohvbd_db(ids)
```

---

ohvbd_dryrun *Option: dry runs of ohvbd searches*

Description

Set this option to make ohvbd terminate searches before execution and return the request object instead.

Note

This is usually only useful when debugging, testing, or developing ohvbd.

Author(s)

Francis Windram

Examples

```
options(ohvbd_dryrun = TRUE)
search_hub("Ixodes ricinus")

options(ohvbd_dryrun = NULL)  # Unset dryrun
```

search_hub *Search vbdhub.org*

Description

Retrieve the IDs for any datasets matching the given search parameters.

Usage

```
search_hub(
  query = "",
  db = c("vt", "vd", "gbif", "px"),
  fromdate = NULL,
  todate = NULL,
  locationpoly = NULL,
  taxonomy = NULL,
  exact = FALSE,
  withoutpublished = TRUE,
  returnlist = FALSE,
  simplify = TRUE,
  connections = 8,
  base_url = "https://api.vbdhub.org"
)
```

Arguments

query	a search string.
db	the databases to search.
fromdate	the date from which to search (ISO format: yyyy-mm-dd).
todate	the date up to which to search (ISO format: yyyy-mm-dd).
locationpoly	a polygon or set of polygons in terra::SpatVector or WKT MULTIPOLYGON format within which to search. Easily generated using match_countries()
taxonomy	a numeric vector containing the gbif ids of taxa to search for (found using match_species() or similar functions).
exact	whether to return exact matches only.
withoutpublished	whether to return results without a publishing date when filtering by date.
returnlist	return the raw output list rather than a formatted dataframe.
simplify	if only a single database was searched, return an ohvbd.ids object instead (defaults to TRUE).
connections	the number of connections to use to parallelise queries.
base_url	the api URL for vbdhub.org (mostly just for QA testing).

Value

an ohvbd.hub.search dataframe, an ohvbd.ids vector (if returnlist=TRUE and length(db) == 1) a list (if returnlist=TRUE) containing the search results.

Author(s)

Francis Windram

Examples

```
search_hub("Ixodes ricinus")
```

---

search_vd *Search VecDyn by keyword*

Description

Retrieve the IDs for any VecDyn datasets matching the given keywords

Usage

```
search_vd(keywords, basereq = vb_basereq())
```

Arguments

keywords	either a string of search terms separated by spaces, or a vector of keywords.
basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

An ohvbd.ids vector of VecDyn dataset IDs.

Note

search_hub() is now preferred for keyword searches:

```
# old style
search_vd(c("Ixodes", "ricinus")

# new style
search_hub("Ixodes ricinus", db = "vd")
```

search_vd() may be deprecated in the future.

Author(s)

Francis Windram

Examples

```
search_vd("Aedes aegypti")

search_vd(c("Aedes", "aegypti"))
```

---

search_vd_smart *Search VecDyn using the explorer's filters*

Description

Retrieve the IDs for any VecDyn datasets matching the given filter.

Usage

```
search_vd_smart(field, operator, value, basereq = vb_basereq())
```

Arguments

field	a field of VecDyn to search.
operator	an operator to use when searching.
value	the value that the field might/might not be.
basereq	an httr2 request object, as generated by vb_basereq(). If NA, uses the default request.

Value

An ohvbd.ids vector of VecDyn dataset IDs.

Valid fields

The following field names are valid (shortcut names are listed in brackets):

- SpeciesName (*species*)
- Title
- Collections
- Years (*yrs*)
- CollectionMethods (*methods*)
- Tags

Valid operators

The following operators are valid (alternative names are listed in brackets):

- `contains` (*contain, has, have*)
- `!contains` (*!contains, !has, !have, ncontains*)
- `equals` (*=, ==, equal, eq*)
- `!equals` (*!=, not, !equal, !eq, neq*)
- `starts` (*starts with, start with, start, sw*)
- `!starts` (*not starts with, not start with, !start, nsw*)
- `in` (*within*)
- `!in` (*not in, not within, !within, nin*)
- `greater` (*greater than, gt, >*)
- `less` (*less than, lt, <*)

Author(s)

Francis Windram

Examples

```
search_vd_smart("Collections", "gt", "1000")
```

---

search_vt *Search VecTraits by keyword*

Description

Retrieve the IDs for any VecTraits datasets matching the given keywords.

Usage

```
search_vt(keywords, basereq = vb_basereq())
```

Arguments

`keywords`	either a string of search terms separated by spaces, or a vector of keywords.
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

An `ohvbd.ids` vector of VecTraits dataset IDs.

Warning

The ids returned from the server (and thus this function) do not necessarily precisely match the keywords that were requested.

For example `search_vt("United Kingdom")` does not return only items found in the United Kingdom. Instead it returns items where some part of the string "United Kingdom" appears in one of the indexed columns.

The indexed columns of VecTraits are:

- `DatasetID`
- `OriginalTraitName`
- `Variables`
- `Interactor1Order`
- `Interactor1Family`
- `Interactor1Genus`
- `Interactor1Species`
- `Interactor1Stage`
- `Interactor1Sex`
- `Interactor2Genus`
- `Interactor2Species`
- `Citation`
- `DOI`
- `CuratedByDOI`
- `SubmittedBy`

Note

`search_hub()` is now preferred for keyword searches:

```
# old style
search_vt(c("Ixodes", "ricinus")

# new style
search_hub("Ixodes ricinus", db = "vt")
```

`search_vt()` may be deprecated in the future.

Author(s)

Francis Windram

Examples

```
search_vt("Aedes aegypti")

search_vt(c("Aedes", "aegypti"))
```

`search_vt_smart` *Search VecTraits using the explorer's filters*

Description

Retrieve the IDs for any VecTraits datasets matching the given filter.

Usage

```
search_vt_smart(field, operator, value, basereq = vb_basereq())
```

Arguments

`field`	a field of VecTraits to search.
`operator`	an operator to use when searching.
`value`	the value that the field might/might not be.
`basereq`	an httr2 request object, as generated by `vb_basereq()`. If `NA`, uses the default request.

Value

An `ohvbd.ids` vector of VecTraits dataset IDs.

Valid fields

The following field names are valid (shortcut names are listed in brackets):

- `DatasetID` (*id*)
- `OriginalTraitName` (*traitname*)
- `Variables`
- `Interactor1Order` (*order*)
- `Interactor1Family` (*family*)
- `Interactor1Genus` (*genus*)
- `Interactor1Species` (*species, spp*)
- `Interactor1Stage` (*stage*)
- `Interactor1Sex` (*sex*)
- `Interactor2Genus` (*genus2*)
- `Interactor2Species` (*species2, spp2*)
- `Citation` (*cite*)
- `DOI`
- `CuratedByDOI` (*curateddoi*)
- `SubmittedBy` (*who*)
- `Tags`

Valid operators

The following operators are valid (alternative names are listed in brackets):

- `contains` (*contain, has, have*)
- `!contains` (*!contains, !has, !have, ncontains*)
- `equals` (`=`, `==`, *equal, eq*)
- `!equals` (*!=, not, !equal, !eq, neq*)
- `starts` (*starts with, start with, start, sw*)
- `!starts` (*not starts with, not start with, !start, nsw*)
- `in` (*within*)
- `!in` (*not in, not within, !within, nin*)

Author(s)

Francis Windram

Examples

```
search_vt_smart("Interactor1Genus", "equals", "Anopheles")
```

---

set_default_ohvbd_cache

Set the default ohvbd cache location

Description

Set the default ohvbd cache location

Usage

```
set_default_ohvbd_cache(d = NULL)
```

Arguments

`d`	The directory to set the cache path to (or NULL to use a default location).

Value

The path of the cache (invisibly)

Note

To permanently set a path to use, add the following to your `.Rprofile` file:

```
options(ohvbd_cache = "path/to/directory")
```

Where `path/to/directory` is the directory in which you wish to cache ohvbd files.

You can find a good default path by running `set_default_ohvbd_cache()` with no arguments.

Author(s)

Francis Windram

Examples

```
set_default_ohvbd_cache()
```

---

set_ohvbd_compat *Set ohvbd compatability mode to TRUE*

Description

Set ohvbd to disable ssl verification for calls to external APIs. This should not be needed (and not be performed) unless you are otherwise experiencing SSL issues when using the package!

When in interactive mode, checks with you to make sure you want to do this. Does not check when run in a script.

Usage

```
set_ohvbd_compat(value = TRUE)
```

Arguments

`value`	The boolean value to set ohvbd_compat to.

Value

No return value, called for side effects

Author(s)

Francis Windram

Examples

```
set_ohvbd_compat()
```

tee *Tee a pipeline to extract the data at a given point*

Description

Add a tee to a pipeline to get the data coming in through the pipe.

This is generally a useful function for debugging pipelines, and for caching data after expensive calls. It is also useful if you want the flexibility of multiple calls with the convenience of a fully-piped approach.

The name `tee` comes from the `tee` shell command within unix systems.

Usage

```
tee(x, .name = "teeout", .env = NULL)
```

Arguments

`x`	The data coming in (whatever it may be).
`.name`	The name to assign to the output within `.env`.
`.env`	The environment within which to save the output at this point. Defaults to the caller env (i.e the env which the pipeline is in).

Value

The value that came from the left hand side of the pipe.

Note

`tee()` does modify the external environment (if `.env` is not specified). This can lead to unpredictable behaviour if not carefully managed, so it is generally worthwhile restricting usage to interactive situations where the environment can be more carefully monitored.

Author(s)

Francis Windram

Examples

```
pipeout <- 1:5 |> exp() |> tee("teeout") |> log()
print(pipeout)
print(teeout)

myenv <- new.env()
pipeout <- 1:5 |> exp() |> tee("teeout", .env = myenv) |> log()
print(myenv$teeout)
```

vb_basereq *Generate a base request object for the vectorbyte databases*

Description

This request is used as the basis for all calls to the vectorbyte API. It does not contain any tokens or session ids, and thus can be regenerated at any time.

Usage

```
vb_basereq(
  baseurl = "https://vectorbyte.crc.nd.edu/portal/api/",
  useragent = "ROHVBD",
  unsafe = FALSE,
  .qa = FALSE
)
```

Arguments

baseurl	the base url for the vectorbyte API.
useragent	the user agent string used when contacting vectorbyte.
unsafe	disable ssl verification (shouldn't ever be required unless you are otherwise experiencing SSL issues!)
.qa	switch to the vb qa server (only useful for testing).

Value

Returns an httr2 request object, pointing at baseurl using useragent.

Author(s)

Francis Windram

Examples

```
basereq <- vb_basereq(
  baseurl="https://vectorbyte.crc.nd.edu/portal/api/",
  useragent="ROHVBD")
```

Index