

Package: publicdataau (via r-universe)

October 15, 2026

Title Australian Government Open Data from 'publicdata.au'

Version 0.4.1

Description Search, filter, total and download the Australian government open data republished at [<https://publicdata.au/>](https://publicdata.au/), where every version of a dataset is kept at a URL that never changes. Rows and totals come from the site's query API as typed, labelled 'tibble' data frames, and whole tables from its files in any of twelve formats. Map layers come as 'sf' objects, and rows that carry the code of an area join to the Australian Bureau of Statistics boundaries in one call. The 'DuckDB' file of any version can be attached over the web and queried in place with 'DBI' or 'dplyr'. Downloads can be kept in a cache, since a version never changes, the changes between versions are listed row by row, and the source file, fetch time and checksum of every version are on record. Every answer carries the version it came from and the attribution the publisher's licence requires, and citations come as 'BibTeX'. Any dataset the site serves works by its slug, so a new dataset needs no new release.

License MIT + file LICENSE

Language en-GB

URL <https://publicdata.au/>

Encoding UTF-8

Depends R (>= 4.1)

Imports httr2 (>= 1.0.0), jsonlite, rlang, tibble

Suggests arrow, DBI, dbplyr, dplyr, duckdb, knitr, rmarkdown, sf, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Cameron Young [aut, cre], National Digital [cph, fnd]

Maintainer Cameron Young <cameron@nationaldigital.com.au>
Config/pak/sysreqs libssl-dev
Repository https://x5hwuk7nc6jsudcc.r-universe.dev
Date/Publication 2026-10-14 12:32:07 UTC
RemoteUrl https://github.com/cran/publicdataau
RemoteRef HEAD
RemoteSha e539984758fd666490112b7c8bddf5854d2634f8

Contents

cache	3
filters	4
pd_aggregate	5
pd_attribution	6
pd_available	6
pd_boundaries	7
pd_boundary_layers	8
pd_browse	8
pd_catalogue	9
pd_changes	10
pd_cite	11
pd_connect	12
pd_dataset	13
pd_datasets	13
pd_download	14
pd_fields	15
pd_join_boundaries	16
pd_latest	17
pd_provenance	18
pd_read	18
pd_rows	19
pd_schema	20
pd_sf	21
pd_tables	22
pd_tbl	22
pd_url	23
pd_versions	24
Index	25

cache	<i>The download cache</i>
-------	---------------------------

Description

A version of a dataset never changes once published, so a file downloaded once can be kept and read again without another download. Nothing is kept unless you ask: pass `'cache = TRUE'` to `[pd_read()]`, `[pd_download()]`, `[pd_connect()]`, `[pd_tbl()]` or `[pd_sf()]`, or set `'options(publicdataau.cache = TRUE)'` for every call.

Usage

```
pd_cache_dir()

pd_cache_list(slug = NULL, version = NULL)

pd_cache_clear(slug = NULL, version = NULL)
```

Arguments

slug	Only this dataset's files. Every dataset when <code>'NULL'</code> .
version	Only this version's files. Every version when <code>'NULL'</code> .

Details

`'pd_cache_dir()'` is where kept files go: `'tools::R_user_dir("publicdataau", "cache")'` unless `'options(publicdataau.cache_dir = )'` names another folder. `'pd_cache_list()'` lists what is kept and `'pd_cache_clear()'` deletes it.

Value

`'pd_cache_dir()'` returns the folder's path. `'pd_cache_list()'` returns a tibble with one row per kept file: its dataset, version, file, size in bytes and when it was saved. `'pd_cache_clear()'` returns the number of bytes freed, invisibly.

See Also

Other files: [pd_download\(\)](#), [pd_read\(\)](#), [pd_url\(\)](#)

Examples

```
pd_cache_dir()
pd_cache_list()
```

filters

*Filters for `pd_rows()` and `pd_aggregate()`***Description**

Each helper builds one condition in the query API's 'operator.value' form. A plain value passed to `[pd_rows()]` must match exactly, a vector of several values matches any of them, and 'NA' matches a blank or suppressed cell.

Usage`pd_eq(value)``pd_neq(value)``pd_gt(value)``pd_gte(value)``pd_lt(value)``pd_lte(value)``pd_like(pattern)``pd_ilike(pattern)``pd_in(...)``pd_is_null()``pd_not(filter)`**Arguments**

value	A value to compare with. Numbers and dates compare as numbers and dates, text as text.
-------	--

pattern	A pattern in which '*' stands for any run of characters.
---------	--

...	Values, any of which may match. None may contain a comma.
-----	---

filter	A filter to negate.
--------	---------------------

Value

A 'pd_filter' object.

See Also

Other query: [pd_aggregate\(\)](#), [pd_connect\(\)](#), [pd_rows\(\)](#), [pd_tbl\(\)](#)

Examples

```
pd_gte(2020)
pd_in("QLD", "NSW")
pd_not(pd_is_null())
```

pd_aggregate	<i>Totals of a dataset by group</i>
--------------	-------------------------------------

Description

Totals of a dataset by group

Usage

```
pd_aggregate(slug, group = NULL, metric = "count", ..., .version = NULL)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
group	Fields to group by.
metric	“count”, “sum.<field>”, “avg.<field>”, “min.<field>” or “max.<field>”, or several of them.
...	Conditions, as for [pd_rows()].
.version	A version date from [pd_versions()]. Without it the answer comes from the newest version and changes when the publisher releases again; with it the answer never changes.

Value

A tibble with one row per group, typed and carrying the same “publicdata” attribute as [pd_rows()].

See Also

Other query: [filters](#), [pd_connect\(\)](#), [pd_rows\(\)](#), [pd_tbl\(\)](#)

Examples

```
pd_aggregate("au-road-deaths", group = "state", year = 2025)
```

pd_attribution	<i>The attribution an answer carries</i>
----------------	--

Description

The attribution an answer carries

Usage

pd_attribution(x)

Arguments

x A data frame from [pd_rows()], [pd_aggregate()], [pd_read()] or [pd_sf()], or a connection from [pd_connect()].

Value

The attribution string the publisher’s licence requires, or ‘NULL’.

See Also

Other metadata: [pd_changes\(\)](#), [pd_cite\(\)](#), [pd_fields\(\)](#), [pd_latest\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
pd_attribution(pd_rows("au-road-deaths", .limit = 1))
```

pd_available	<i>Is publicdata.au reachable?</i>
--------------	------------------------------------

Description

Asks the site for its catalogue’s headers, with a short timeout. Use it to skip work that needs the site when there is no connection.

Usage

pd_available()

Value

‘TRUE’ when the site answers, else ‘FALSE’. It never fails.

See Also

Other find: [pd_browse\(\)](#), [pd_catalogue\(\)](#), [pd_dataset\(\)](#), [pd_datasets\(\)](#), [pd_versions\(\)](#)

Examples

```
pd_available()
```

pd_boundaries	<i>A boundary layer as an sf object</i>
---------------	---

Description

A boundary layer as an sf object

Usage

```
pd_boundaries(layer, cache = NULL)
```

Arguments

layer	A layer's key, such as "lga", "sa2", "suburb", "postcode", "state_electorate" or "federal_electorate", or its slug or code field. <code>[pd_boundary_layers()]</code> lists them.
cache	Keep the file in <code>[pd_cache_dir()]</code> and reuse it next time. Versions never change, so a kept file never goes stale. Off unless <code>'options(publicdataau.cache = TRUE)'</code> is set.

Value

An 'sf' tibble with a row per area: its code, name and the other fields of the layer, in GDA2020 (EPSG:7844).

See Also

Other maps: [pd_boundary_layers\(\)](#), [pd_join_boundaries\(\)](#), [pd_sf\(\)](#)

Examples

```
# A boundary file is tens of megabytes, so this runs only when asked.
electorates <- pd_boundaries("federal_electorate")
```

pd_boundary_layers	<i>Boundary layers</i>
--------------------	------------------------

Description

The ABS boundaries publicdata.au serves, each keyed by a code that rows of other datasets carry: council areas (LGA), SA2s, suburbs, postal areas and state and federal electorates.

Usage

```
pd_boundary_layers()
```

Value

A tibble with one row per layer: its ‘key’ (such as “lga”), ‘slug’, ‘title’, the ‘code’ and ‘name’ fields that identify an area, a ‘noun’ for one area, and the ‘version’ served.

See Also

Other maps: [pd_boundaries\(\)](#), [pd_join_boundaries\(\)](#), [pd_sf\(\)](#)

Examples

```
pd_boundary_layers()
```

pd_browse	<i>Open a dataset’s page</i>
-----------	------------------------------

Description

Opens the dataset’s page, or one version’s, in the browser, in an interactive session.

Usage

```
pd_browse(slug, version = NULL)
```

Arguments

- slug The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
- version A version date from [pd_versions()]. The newest when ‘NULL’.

Value

The page’s URL, invisibly.

See Also

Other find: [pd_available\(\)](#), [pd_catalogue\(\)](#), [pd_dataset\(\)](#), [pd_datasets\(\)](#), [pd_versions\(\)](#)

Examples

```
if (interactive()) pd_browse("au-road-deaths")
```

pd_catalogue	<i>Search every dataset on the government portals</i>
--------------	---

Description

publicdata.au reads the catalogues of the Australian government open data portals, well over a hundred thousand datasets, of which it serves a small part. This searches all of them, so a dataset that is not served yet can be found, and says which are served. Vote for one to be built on its page.

Usage

```
pd_catalogue(
  q = NULL,
  jurisdiction = NULL,
  status = NULL,
  limit = 20,
  offset = 0
)
```

Arguments

q	Words to search titles, summaries and publishers for. The newest records when 'NULL'.
jurisdiction	A government, as a code such as "qld" or a name such as "Queensland".
status	Any of "served" (on publicdata.au), "votable" (can be voted for), "chosen" (being built) and "closed" (cannot be built, with the reason).
limit	Records to return, 1 to 50.
offset	Records to skip, for the next page.

Value

A tibble with one row per record: its 'id', 'title', 'summary', 'publisher', 'jurisdiction', portal 'url', 'licence', 'formats', 'modified', 'status', the publicdata.au 'page' when served and the 'reason' when closed. 'attr(x, "total")' is the number of matches.

See Also

Other find: [pd_available\(\)](#), [pd_browse\(\)](#), [pd_dataset\(\)](#), [pd_datasets\(\)](#), [pd_versions\(\)](#)

Examples

```
pd_catalogue("water quality", jurisdiction = "Queensland", limit = 5)
```

pd_changes	<i>What changed between versions</i>
------------	--------------------------------------

Description

Each time a publisher releases again, the site compares the new version with the one before it, row by row on the dataset’s key. ‘pd_changes()’ lists those comparisons; ‘pd_diff()’ reads one of them in full.

Usage

```
pd_changes(slug, from = NULL, to = NULL)

pd_diff(slug, version = NULL)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
from, to	Version dates bounding the comparisons listed. Every comparison kept when both are ‘NULL’.
version	The newer version of the pair: the comparison with the version before it. The newest when ‘NULL’.

Value

‘pd_changes()’ returns a tibble with one row per comparison, oldest first: the two versions, their rows, how many rows were added, removed, changed and unchanged, the fields added and removed, and the comparison’s URL. A database is compared by each table’s row count, so its added, removed and changed are ‘NA’ and its tables added and removed are listed as fields.

‘pd_diff()’ returns the comparison as a list of class ‘pd_diff’: the counts above, the keys of the rows added, removed and changed (up to 50,000 of each, with ‘truncated’ saying when there were more) and up to ten changed rows with each changed field’s old and new value.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_cite\(\)](#), [pd_fields\(\)](#), [pd_latest\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
pd_changes("rba-money-market-daily")

steps <- pd_changes("rba-money-market-daily")
if (nrow(steps)) pd_diff("rba-money-market-daily", steps$to[nrow(steps)])
```

pd_cite

*How to cite a version***Description**

A citation for one version of a dataset, naming the publisher, the version and the URL it is kept at, with the attribution the publisher's licence requires. Print it for text, or pass it to [utils::toBibtex()] for a reference manager.

Usage

```
pd_cite(slug, version = NULL)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from [pd_versions()]. The newest when 'NULL'.

Value

A [utils::bibentry()] object.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_changes\(\)](#), [pd_fields\(\)](#), [pd_latest\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
cit <- pd_cite("rba-cash-rate")
print(cit, style = "text")
utils::toBibtex(cit)
```

pd_connect

*Connect to a version with DuckDB***Description**

Attaches a version's DuckDB file read-only over HTTPS and makes it the current database, so every table and view can be queried by name with 'DBI' or 'dplyr'. Only the blocks a query touches are read, so a count over millions of rows runs without a download. For a database such as G-NAF the file holds every table, the keys between them and the publisher's views; for a single table it holds 'records'.

Usage

```
pd_connect(
  slug,
  version = NULL,
  name = NULL,
  shared_home = FALSE,
  cache = NULL,
  ...
)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from [pd_versions()]. The newest when 'NULL'.
name	The name the file is attached as. The slug with hyphens as underscores when 'NULL'.
shared_home	Where 'DuckDB' keeps the 'httpfs' extension it downloads to read over HTTPS. 'FALSE', the default, keeps it in the session's temporary directory, so it is downloaded again in a new session. 'TRUE' keeps it under '~/duckdb' for every session.
cache	Download the whole file into [pd_cache_dir()] once and attach it from there, which makes repeated scans of a large database much faster. Off unless 'options(publicdataau.cache = TRUE)' is set.
...	Passed to [duckdb::duckdb()].

Value

A 'DBI' connection. 'attr(con, "publicdata")' names the dataset, the version, the file's URL and local path if cached, and carries the file's provenance: licence, attribution and citation. Disconnect with 'DBI::dbDisconnect(con)'.

See Also

Other query: [filters](#), [pd_aggregate\(\)](#), [pd_rows\(\)](#), [pd_tbl\(\)](#)

Examples

```
con <- pd_connect("au-road-deaths")
DBI::dbGetQuery(con, "SELECT state, count(*) AS n FROM records GROUP BY 1")
DBI::dbDisconnect(con)
```

pd_dataset	<i>A dataset's description and files</i>
------------	--

Description

A dataset's description and files

Usage

```
pd_dataset(slug)
```

Arguments

slug The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.

Value

The dataset's Frictionless data package as a list: title, licence, attribution, fields and every file of the newest version.

See Also

Other find: [pd_available\(\)](#), [pd_browse\(\)](#), [pd_catalogue\(\)](#), [pd_datasets\(\)](#), [pd_versions\(\)](#)

Examples

```
pd_dataset("au-road-deaths")$title
```

pd_datasets	<i>Datasets served by publicdata.au</i>
-------------	---

Description

Datasets served by publicdata.au

Usage

```
pd_datasets(q = NULL, publisher = NULL, topic = NULL, jurisdiction = NULL)
```

Arguments

q	Words to search titles, summaries, publishers, keywords and field names for. Every served dataset when 'NULL'.
publisher	Part of a publisher's name, such as "Bureau of Meteorology" or "Transport", ignoring case.
topic	A topic the site files datasets under, such as "roads", "crime" or "housing". An unknown topic fails with the list of topics.
jurisdiction	The government a publisher belongs to, as a code such as "Qld" or "Cth" or a name such as "Queensland", ignoring case.

Value

A tibble with one row per dataset: its slug, title, publisher, licence, page and newest Parquet file. Every condition given must match.

See Also

Other find: [pd_available\(\)](#), [pd_browse\(\)](#), [pd_catalogue\(\)](#), [pd_dataset\(\)](#), [pd_versions\(\)](#)

Examples

```
pd_datasets("road crashes")
pd_datasets(publisher = "Transport", jurisdiction = "Queensland")
```

pd_download	<i>Download one version's file</i>
-------------	------------------------------------

Description

Download one version's file

Usage

```
pd_download(
  slug,
  format = "parquet",
  version = NULL,
  path = NULL,
  table = NULL,
  cache = NULL
)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
format	One of "parquet", "csv", "csv.gz", "json", "ndjson", "sqlite", "duckdb", "xlsx", "arrow", "geojson", "gpkg" or "geo.parquet". The last three are served only for datasets with a location or a shape.
version	A version date from [pd_versions()]. The newest when 'NULL'.
path	Where to save the file. When 'NULL', a file in the session's temporary directory named for the slug and version, or the file in the cache when 'cache' is on.
table	For a database, one of its tables, which is served as Parquet. See [pd_tables()].
cache	Keep the file in [pd_cache_dir()] and reuse it next time. Versions never change, so a kept file never goes stale. Off unless 'options(publicdataau.cache = TRUE)' is set.

Value

The path of the saved file, invisibly, with the version it resolved to in 'attr(path, "version")'.

See Also

Other files: [cache](#), [pd_read\(\)](#), [pd_url\(\)](#)

Examples

```
f <- pd_download("qld-road-crash-factors", "csv")
```

pd_fields	<i>A dataset's fields</i>
-----------	---------------------------

Description

What each field holds: its name, type and the publisher's description, with the range of a number or date and the values of a field that holds few. Read it before filtering with [pd_rows()], since a condition must name a field exactly and match its values exactly.

Usage

```
pd_fields(slug, version = NULL)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from [pd_versions()]. The newest when 'NULL'.

Value

A tibble with one row per field: ‘name’, ‘type’ (“string”, “integer”, “number”, “boolean”, “date” or “datetime”), ‘description’, ‘min’ and ‘max’ as text, and ‘values’, a list of the values a field holds when it holds few. For a database such as G-NAF there is a row per field of every table, with a ‘table’ column, and no ranges. Without ‘version’ the fields are the newest version’s; a pinned version is described by its schema, without ranges or values.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_changes\(\)](#), [pd_cite\(\)](#), [pd_latest\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
pd_fields("au-road-deaths")
```

pd_join_boundaries	<i>Join rows to their boundaries</i>
--------------------	--------------------------------------

Description

Adds the boundary of the area each row names, by its ABS code, and returns an ‘sf’ object ready to map. Rows from [pd_rows()] or [pd_aggregate()] carry the codes of the areas their point falls in, such as ‘lga_2025_code’, and the layer is found from the column’s name; any data frame with a column of codes works with ‘layer’ and ‘by’. Aggregate first: a map of 500 council areas is quicker to draw than 80,000 rows each carrying its area’s shape.

Usage

```
pd_join_boundaries(x, layer = NULL, by = NULL, cache = NULL)
```

Arguments

- | | |
|-------|--|
| x | A data frame. |
| layer | The layer to join to, as for [pd_boundaries()]. Found from the column names of ‘x’ when ‘NULL’. |
| by | The column of ‘x’ that holds the codes. The layer’s code field, such as “lga_2025_code”, when ‘NULL’. Numeric codes are compared as text, with leading zeros restored, so postcode 800 matches “0800”. |
| cache | Keep the file in [pd_cache_dir()] and reuse it next time. Versions never change, so a kept file never goes stale. Off unless ‘options(publicdataau.cache = TRUE)’ is set. |

Value

An 'sf' tibble with the rows of 'x' in their order, the area's name when 'x' lacks it, and the area's geometry. A row whose code matches no area has an empty geometry, and a message says how many.

See Also

Other maps: [pd_boundaries\(\)](#), [pd_boundary_layers\(\)](#), [pd_sf\(\)](#)

Examples

```
# Downloads the SA2 boundaries, tens of megabytes, so this runs only when asked.
by_sa2 <- pd_aggregate("act-road-crashes", group = "sa2_2021_code")
map <- pd_join_boundaries(by_sa2)
plot(map["count"])
```

pd_latest

The newest version of a dataset

Description

The newest version of a dataset

Usage

```
pd_latest(slug)
```

Arguments

slug The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.

Value

The newest version's date as text, such as '"2026-09-28"'.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_changes\(\)](#), [pd_cite\(\)](#), [pd_fields\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
pd_latest("au-road-deaths")
```

pd_provenance	<i>Where a version came from</i>
---------------	----------------------------------

Description

The record kept with every version: the publisher's file it was read from, with its address, name, size and SHA-256, when it was fetched, the licence as the publisher stated it and when that was read, and the rows and fields it holds. It answers "where did this number come from" for a reviewer.

Usage

```
pd_provenance(slug, version = NULL)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from [pd_versions()]. The newest when 'NULL'.

Value

A list of class 'pd_provenance', the version's 'manifest.json'.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_changes\(\)](#), [pd_cite\(\)](#), [pd_fields\(\)](#), [pd_latest\(\)](#), [pd_schema\(\)](#), [pd_tables\(\)](#)

Examples

```
pd_provenance("au-road-deaths")
```

pd_read	<i>Read a whole table</i>
---------	---------------------------

Description

Downloads one version's Parquet file, or one table of a database, and reads it with 'arrow'.

Usage

```
pd_read(slug, version = NULL, table = NULL, cache = NULL, columns = NULL)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from [pd_versions()]. The newest when 'NULL'.
table	For a database, one of its tables, which is served as Parquet. See [pd_tables()].
cache	Keep the file in [pd_cache_dir()] and reuse it next time. Versions never change, so a kept file never goes stale. Off unless 'options(publicdataau.cache = TRUE)' is set.
columns	Fields to read, which saves memory on a wide table. Every field when 'NULL'.

Value

A tibble, each column labelled with its field's description. 'attr(x, "publicdata")' is the provenance header the file itself carries: version, licence, attribution, citation and source.

See Also

Other files: [cache](#), [pd_download\(\)](#), [pd_url\(\)](#)

Examples

```
crashes <- pd_read("qld-road-crash-factors")
```

pd_rows

Rows of a dataset

Description

Reads rows from the query API. For a whole table, [pd_read()] or [pd_download()] is faster and has no rate limit.

Usage

```
pd_rows(
  slug,
  ...,
  .select = NULL,
  .order = NULL,
  .limit = NULL,
  .offset = NULL,
  .version = NULL,
  .all = FALSE
)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
...	Conditions, each a field name and a value or a filter such as ‘year = pd_gte(2020)’. A vector of several values matches any of them and ‘NA’ matches a blank or suppressed cell. Every condition must match.
.select	Fields to return. Every field when ‘NULL’.
.order	Fields to sort by, each “field.asc” or “field.desc”.
.limit	Rows per page, 1 to 10000.
.offset	Rows to skip.
.version	A version date from [pd_versions()]. Without it the answer comes from the newest version and changes when the publisher releases again; with it the answer never changes.
.all	Follow every page and return all matching rows.

Value

A tibble, with each column typed as its field says (dates as ‘Date’, booleans as logical) and labelled with the field’s description. ‘attr(x, “publicdata”)’ holds the version, licence and attribution the answer came with; see [pd_attribution()].

See Also

Other query: [filters](#), [pd_aggregate\(\)](#), [pd_connect\(\)](#), [pd_tbl\(\)](#)

Examples

```
pd_rows("au-road-deaths", state = "QLD", year = pd_gte(2024), .limit = 5)
```

pd_schema	<i>A version’s schema</i>
-----------	---------------------------

Description

A version’s schema

Usage

```
pd_schema(slug, version = NULL)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
version	A version date from [pd_versions()]. The newest when ‘NULL’.

Value

The version's 'schema.json' as a list: the fields of a table, or for a database every table with its fields, keys and references, and the views. `[pd_fields()]` gives the fields as a tibble.

See Also

Other metadata: `pd_attribution()`, `pd_changes()`, `pd_cite()`, `pd_fields()`, `pd_latest()`, `pd_provenance()`, `pd_tables()`

Examples

```
names(pd_schema("qld-road-crash-factors"))
```

pd_sf

*Read a dataset's map layer as an sf object***Description**

Downloads one version's GeoPackage and reads it with 'sf'. Datasets with a location, such as crash points, or a shape, such as ABS boundaries, have one; their coordinates are in the reference system the publisher used, which the GeoPackage names, usually GDA2020 (EPSG:7844).

Usage

```
pd_sf(slug, version = NULL, cache = NULL)
```

Arguments

slug	The dataset's slug, as in its page URL 'https://publicdata.au/d/<slug>/'.
version	A version date from <code>[pd_versions()]</code> . The newest when 'NULL'.
cache	Keep the file in <code>[pd_cache_dir()]</code> and reuse it next time. Versions never change, so a kept file never goes stale. Off unless 'options(publicdataau.cache = TRUE)' is set.

Value

An 'sf' tibble with a row per feature. 'attr(x, "publicdata")' is the provenance the file carries: version, licence, attribution, citation and source.

See Also

Other maps: `pd_boundaries()`, `pd_boundary_layers()`, `pd_join_boundaries()`

Examples

```
storages <- pd_sf("au-water-storages")
sf::st_crs(storages)$epsg
```

pd_tables	<i>The tables of a dataset</i>
-----------	--------------------------------

Description

A database such as G-NAF is served as one DuckDB file holding several tables, with one Parquet file per table beside it. A dataset that is one table has one entry, ‘records’.

Usage

```
pd_tables(slug, version = NULL)
```

Arguments

- slug The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
- version A version date from [pd_versions()]. The newest when ‘NULL’.

Value

A tibble with one row per table: its name, description, rows, number of fields and key.

See Also

Other metadata: [pd_attribution\(\)](#), [pd_changes\(\)](#), [pd_cite\(\)](#), [pd_fields\(\)](#), [pd_latest\(\)](#), [pd_provenance\(\)](#), [pd_schema\(\)](#)

Examples

```
pd_tables("qld-road-crash-factors")
```

pd_tbl	<i>A table to query with dplyr</i>
--------	------------------------------------

Description

Attaches a version’s DuckDB file with [pd_connect()] and returns one of its tables or views as a lazy ‘dplyr’ table. Verbs such as ‘filter()’, ‘group_by()’ and ‘summarise()’ are translated to SQL and run by ‘DuckDB’ over HTTPS, reading only the blocks they need; ‘collect()’ brings the result into R. Calls for the same dataset and version share one connection.

Usage

```
pd_tbl(slug, table = NULL, version = NULL, cache = NULL)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
table	The table or view to query. A dataset that is one table has one, ‘records’, which is the default; for a database such as G-NAF, name one of [pd_tables()] or a view such as “address_view”.
version	A version date from [pd_versions()]. The newest when ‘NULL’.
cache	Download the whole file into [pd_cache_dir()] once and attach it from there, which makes repeated scans of a large database much faster. Off unless ‘options(publicdataau.cache = TRUE)’ is set.

Value

A lazy table from ‘dbplyr’. Its connection is ‘dbplyr::remote_con(x)’, which carries the same “publicdata” attribute as [pd_connect()].

See Also

Other query: [filters](#), [pd_aggregate\(\)](#), [pd_connect\(\)](#), [pd_rows\(\)](#)

Examples

```
library(dplyr)
pd_tbl("au-road-deaths") |>
  filter(year >= 2020) |>
  count(state) |>
  collect()
```

pd_url	<i>A file’s URL</i>
--------	---------------------

Description

The address of one version’s file, for reading with another tool such as ‘read.csv()’, ‘arrow’ or ‘DuckDB’, or for a ‘targets’ pipeline with ‘format = “url”’, which reruns when the address changes.

Usage

```
pd_url(slug, format = "parquet", version = NULL, table = NULL)
```

Arguments

slug	The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.
format	One of “parquet”, “csv”, “csv.gz”, “json”, “ndjson”, “sqlite”, “duckdb”, “xlsx”, “arrow”, “geojson”, “gpkg” or “geo.parquet”. The last three are served only for datasets with a location or a shape.
version	A version date from [pd_versions()]. The newest when ‘NULL’.
table	For a database, one of its tables, which is served as Parquet. See [pd_tables()].

Value

The URL as text. With a ‘version‘ it names a file that never changes; without one it names the newest version’s file, which redirects to a dated URL and moves when the publisher releases again.

See Also

Other files: [cache](#), [pd_download\(\)](#), [pd_read\(\)](#)

Examples

```
pd_url("au-road-deaths", "csv", "2026-08-07")
```

pd_versions	<i>Every version of a dataset</i>
-------------	-----------------------------------

Description

Every version of a dataset

Usage

```
pd_versions(slug)
```

Arguments

slug The dataset’s slug, as in its page URL ‘https://publicdata.au/d/<slug>/’.

Value

A tibble with one row per version, newest first: its date, as-at date, rows, fields and the SHA-256 of the publisher’s file.

See Also

Other find: [pd_available\(\)](#), [pd_browse\(\)](#), [pd_catalogue\(\)](#), [pd_dataset\(\)](#), [pd_datasets\(\)](#)

Examples

```
pd_versions("au-road-deaths")
```

Index

- * **files**
 - cache, [3](#)
 - pd_download, [14](#)
 - pd_read, [18](#)
 - pd_url, [23](#)
- * **find**
 - pd_available, [6](#)
 - pd_browse, [8](#)
 - pd_catalogue, [9](#)
 - pd_dataset, [13](#)
 - pd_datasets, [13](#)
 - pd_versions, [24](#)
- * **maps**
 - pd_boundaries, [7](#)
 - pd_boundary_layers, [8](#)
 - pd_join_boundaries, [16](#)
 - pd_sf, [21](#)
- * **metadata**
 - pd_attribution, [6](#)
 - pd_changes, [10](#)
 - pd_cite, [11](#)
 - pd_fields, [15](#)
 - pd_latest, [17](#)
 - pd_provenance, [18](#)
 - pd_schema, [20](#)
 - pd_tables, [22](#)
- * **query**
 - filters, [4](#)
 - pd_aggregate, [5](#)
 - pd_connect, [12](#)
 - pd_rows, [19](#)
 - pd_tbl, [22](#)

cache, [3](#), [15](#), [19](#), [24](#)

filters, [4](#), [5](#), [12](#), [20](#), [23](#)

pd_aggregate, [5](#)

pd_aggregate(), [5](#), [12](#), [20](#), [23](#)

pd_attribution, [6](#)

pd_attribution(), [10](#), [11](#), [16–18](#), [21](#), [22](#)

pd_available, [6](#)

pd_available(), [9](#), [13](#), [14](#), [24](#)

pd_boundaries, [7](#)

pd_boundaries(), [8](#), [17](#), [21](#)

pd_boundary_layers, [8](#)

pd_boundary_layers(), [7](#), [17](#), [21](#)

pd_browse, [8](#)

pd_browse(), [7](#), [9](#), [13](#), [14](#), [24](#)

pd_cache_clear(cache), [3](#)

pd_cache_dir(cache), [3](#)

pd_cache_list(cache), [3](#)

pd_catalogue, [9](#)

pd_catalogue(), [7](#), [9](#), [13](#), [14](#), [24](#)

pd_changes, [10](#)

pd_changes(), [6](#), [11](#), [16–18](#), [21](#), [22](#)

pd_cite, [11](#)

pd_cite(), [6](#), [10](#), [16–18](#), [21](#), [22](#)

pd_connect, [12](#)

pd_connect(), [5](#), [20](#), [23](#)

pd_dataset, [13](#)

pd_dataset(), [7](#), [9](#), [14](#), [24](#)

pd_datasets, [13](#)

pd_datasets(), [7](#), [9](#), [13](#), [24](#)

pd_diff(pd_changes), [10](#)

pd_download, [14](#)

pd_download(), [3](#), [19](#), [24](#)

pd_eq(filters), [4](#)

pd_fields, [15](#)

pd_fields(), [6](#), [10](#), [11](#), [17](#), [18](#), [21](#), [22](#)

pd_gt(filters), [4](#)

pd_gte(filters), [4](#)

pd_ilike(filters), [4](#)

pd_in(filters), [4](#)

pd_is_null(filters), [4](#)

pd_join_boundaries, [16](#)

pd_join_boundaries(), [7](#), [8](#), [21](#)

pd_latest, [17](#)

pd_latest(), [6](#), [10](#), [11](#), [16](#), [18](#), [21](#), [22](#)

`pd_like(filters)`, 4
`pd_lt(filters)`, 4
`pd_lte(filters)`, 4
`pd_neq(filters)`, 4
`pd_not(filters)`, 4
`pd_provenance`, 18
`pd_provenance()`, 6, 10, 11, 16, 17, 21, 22
`pd_read`, 18
`pd_read()`, 3, 15, 24
`pd_rows`, 19
`pd_rows()`, 5, 12, 23
`pd_schema`, 20
`pd_schema()`, 6, 10, 11, 16–18, 22
`pd_sf`, 21
`pd_sf()`, 7, 8, 17
`pd_tables`, 22
`pd_tables()`, 6, 10, 11, 16–18, 21
`pd_tbl`, 22
`pd_tbl()`, 5, 12, 20
`pd_url`, 23
`pd_url()`, 3, 15, 19
`pd_versions`, 24
`pd_versions()`, 7, 9, 13, 14