Package 'datapond'

November 17, 2026

Title Query Curated 'DuckDB' Databases Built from Public Data

Version 0.1.1

Description Connects to the 'datapond' registry of curated 'DuckDB' databases built from public government and research data (immigration courts, campaign finance, clinical trials, Medicare, and more). Databases are attached remotely over HTTP so only the byte ranges a query touches are transferred, or downloaded once for local use. Returns standard 'DBI' connections that work with 'dbplyr'.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Depends R (>= 4.1)

Imports curl (>= 5.0.0), DBI, duckdb (>= 1.0.0), jsonlite, tools, utils

Suggests callr, httpuv, dbplyr (>= 2.3.0), dplyr, knitr, rmarkdown, testthat (>= 3.0.0), tibble, withr

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/datapond-db/datapond-r, https://datapond-db.github.io/website/

BugReports https://github.com/datapond-db/datapond-r/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Ian Nason [aut, cre]

Maintainer Ian Nason <ign.nason@gmail.com>

Repository CRAN

Date/Publication 2026-11-17 08:00:02 UTC

Contents

dp_connect 2
dp_data_dir 3
dp_describe 4
dp_disconnect 4
dp_download 5
dp_info 6
dp_list 6
dp_local_path 7
dp_registry 7
dp_update 8

Index **9**

`dp_connect` *Connect to one or more datapond databases*

Description

Opens an in-memory DuckDB connection and attaches each requested database read-only, either remotely from its `attach_url` (the default; DuckDB's `httpfs` extension fetches byte ranges without requiring a full download first) or from a local copy made with `dp_download()`. Large remote queries can still transfer much of a file.

Usage

```
dp_connect(
  id,
  local = FALSE,
  quiet = FALSE,
  config = getOption("datapond.duckdb_config", list())
)
```

Arguments

`id`	One or more database ids (see `dp_list()`).
`local`	Attach local copies under `dp_data_dir()` instead of remote files?
`quiet`	Suppress progress messages?
`config`	Named list of DuckDB configuration values, passed to `duckdb::duckdb()` before any database is attached. Defaults to `getOption("datapond.duckdb_config", list())`. Use `memory_limit`, `threads`, `temp_directory`, and `max_temp_directory_size` to bound query resources. Values must be non-missing scalars and are converted to strings for DuckDB. DuckDB's memory limit does not include all R allocations. This argument was introduced in development version 0.1.1 and is not in CRAN 0.1.0. With CRAN 0.1.0, set query limits after connecting using `DBI::dbExecute()`, or use a direct DuckDB connection for startup limits.

Details

With a single `id` the database is made the default catalog (`USE`), so tables can be referenced unqualified. With several ids, qualify tables with the database id, double-quoting ids that contain a hyphen: `"cms-medicare".physician_summary`.

Value

A `duckdb_connection` (a `DBIConnection`). Close it with `dp_disconnect()`.

Examples

```
## Not run:
con <- dp_connect("eoir")
DBI::dbGetQuery(con, "SELECT * FROM proceedings LIMIT 5")
dp_disconnect(con)

con <- dp_connect(c("cms-medicare", "openpayments"))
DBI::dbGetQuery(con, 'SELECT * FROM "cms-medicare".physician_summary LIMIT 5')
dp_disconnect(con)

## End(Not run)
```

`dp_data_dir` *Local directories used by datapond*

Description

The registry is cached under `tools::R_user_dir("datapond", "cache")` and downloaded databases live under `tools::R_user_dir("datapond", "data")`. Override either with `options(datapond.cache_dir = ...)` / `options(datapond.data_dir = ...)`. Setting `options(datapond.data_dir = "~/.datapond")` shares downloads with the Python package, which names local files the same way (`<id>.duckdb`).

Usage

```
dp_data_dir()

dp_cache_dir()
```

Value

A path (character scalar).

Examples

```
dp_data_dir()
```

dp_describe *Explore a database's data dictionary*

Description

Every datapond database ships `_metadata` (one row per table) and `_columns` (types, null rates, example values, join hints). `dp_describe()` reads them and falls back to `information_schema` when they are absent.

Usage

```
dp_describe(x, table = NULL, search = NULL)
```

Arguments

`x`	A database id, or an open connection from `dp_connect()`.
`table`	Describe the columns of this table instead of listing tables.
`search`	Find columns whose name contains this text (case-insensitive).

Value

A data frame (a tibble when the tibble package is installed).

Examples

```
## Not run:
dp_describe("eoir")
dp_describe("eoir", table = "proceedings")
dp_describe("eoir", search = "judge")

## End(Not run)
```

dp_disconnect *Close a datapond connection*

Description

Close a datapond connection

Usage

```
dp_disconnect(con)
```

Arguments

`con`	A connection from `dp_connect()`.

Value

TRUE, invisibly.

---

dp_download *Download a database for local use*

Description

Downloads the full .duckdb file so later queries run at disk speed. Files are saved as <id>.duckdb under dp_data_dir() (or path), which is where dp_connect(id, local = TRUE) looks for them.

Usage

```
dp_download(id, path = NULL, quiet = FALSE, resume = TRUE)
```

Arguments

id	Database id.
path	Destination file or directory. Defaults to dp_data_dir().
quiet	Suppress the progress bar?
resume	Resume a partial download if one exists?

Details

The transfer goes to <file>.part and replaces the destination only after it is complete and opens as a DuckDB database, so an existing copy is never damaged by a failed download. A partial download is resumed only when the server identifies the file's revision (ETag, else Last-Modified), the partial file was started against that revision, and the ranged request is bound to it with If-Range; otherwise it restarts. A sidecar <file>.datapond.json records the remote file identity for dp_update().

Value

The local path, invisibly.

Examples

```
## Not run:
dp_download("dol-visas")
con <- dp_connect("dol-visas", local = TRUE)

## End(Not run)
```

dp_info *Information about one database*

Description

Information about one database

Usage

```
dp_info(id)
```

Arguments

id	Database id.

Value

The registry entry (a list of class `datapond_db`), printed in a readable layout.

Examples

```
## Not run:
dp_info("eoir")

## End(Not run)
```

dp_list *List available databases*

Description

List available databases

Usage

```
dp_list()

dp_databases()
```

Value

`dp_list()` returns a character vector of database ids; `dp_databases()` returns one row per database with its registry fields.

Examples

```
## Not run:
dp_list()
dp_databases()[, c("id", "rows", "size_gb")]

## End(Not run)
```

`dp_local_path` *Path of a locally downloaded database*

Description

Path of a locally downloaded database

Usage

```
dp_local_path(id, must_exist = FALSE)
```

Arguments

`id`	Database id (see `dp_list()`).
`must_exist`	Error if the file is not present?

Value

A path (character scalar).

Examples

```
dp_local_path("eoir")
```

`dp_registry` *The datapond registry*

Description

Fetches `registry.json` from GitHub (cached for an hour under `dp_cache_dir()`; a stale cache is used if the network is unavailable).

Usage

```
dp_registry(refresh = FALSE)
```

Arguments

`refresh`	Ignore the cache and fetch again?

Value

The parsed registry: a list with a `databases` element, one list per database.

Examples

```
## Not run:
reg <- dp_registry()
length(reg$databases)

## End(Not run)
```

`dp_update` *Update a local database if the remote file has changed*

Description

Compares the identity of the remote file (ETag, else Last-Modified, from a HEAD request) with the one recorded when the local copy was downloaded, and re-downloads when they differ or when the revision cannot be verified. Copies without a usable identity record are re-downloaded when update is requested; local modification times and registry dates cannot prove which bytes they contain. The registry is re-fetched first.

Usage

```
dp_update(id, quiet = FALSE)
```

Arguments

`id`	Database id.
`quiet`	Suppress the progress bar?

Value

The local path, invisibly.

Index

DBI::dbExecute(), *2*
dp_cache_dir *(*dp_data_dir*)*, 3
dp_cache_dir(), *7*
dp_connect, 2
dp_connect(), *4*
dp_data_dir, 3
dp_data_dir(), *2*, *5*
dp_databases *(*dp_list*)*, 6
dp_describe, 4
dp_disconnect, 4
dp_disconnect(), *3*
dp_download, 5
dp_download(), *2*
dp_info, 6
dp_list, 6
dp_list(), *2*, *7*
dp_local_path, 7
dp_registry, 7
dp_update, 8
dp_update(), *5*
duckdb::duckdb(), *2*