

Package ‘delta.sharing’

October 6, 2026

Title Client for the Delta Sharing Protocol

Version 0.2.1

Description Connects R to 'Delta Sharing' servers for share discovery, table metadata, snapshot reads, and change data feed reads.

License Apache License (>= 2)

URL <https://zacdav-db.github.io/delta-sharing-r/>,
<https://github.com/zacdav-db/delta-sharing-r>

BugReports <https://github.com/zacdav-db/delta-sharing-r/issues>

Depends R (>= 4.3.0)

Encoding UTF-8

Imports arrow, cli (>= 3.0.0), fs, httr2 (>= 1.2.0), jose, jsonlite,
nanoarrow (>= 0.8.0), openssl, purrr (>= 1.0.0), R6, rlang (>= 1.2.0), tibble

Suggests bit64, DBI, dplyr, duckdb, knitr, rmarkdown, testthat (>= 3.1.7), withr

VignetteBuilder knitr

Config/Needs/check rcmdcheck

Config/Needs/coverage covr

Config/Needs/documentation roxygen2

Config/Needs/website pkgdown

Config/testthat/edition 3

Config/testthat/parallel false

RoxyenNote 7.3.2

SystemRequirements Cargo (Rust's package manager), rustc (>= 1.88)

NeedsCompilation yes

Author Zac Davies [aut, cre],
Databricks [cph]

Maintainer Zac Davies <zac@databricks.com>

Repository CRAN

Date/Publication 2026-10-06 07:40:21 UTC

Contents

demo_profile	2
SharingChanges	2
SharingClient	3
SharingSnapshot	5
SharingTable	6
sharing_client	9

Index	10
-------	----

demo_profile	<i>Delta Sharing open-datasets profile</i>
--------------	--

Description

Fetches the official profile for the Delta Sharing open-datasets server and returns it as a parsed list. The profile is public, but fetching it requires an internet connection.

Usage

```
demo_profile()
```

Value

A parsed Delta Sharing profile list.

Examples

```
profile <- demo_profile()
names(profile)
```

SharingChanges	<i>Delta Sharing change data feed reader</i>
----------------	--

Description

Delta Sharing change data feed reader
Delta Sharing change data feed reader

Details

An immutable change data feed specification with Arrow materializers. Created by `SharingTable$changes()`. The sharing server validates the supplied bounds.

Super class

`delta.sharing::SharingReader` -> SharingChanges

Methods**Public methods:**

- `SharingChanges$new()`

Method `new()`: Create a changes reader. Prefer `SharingTable$changes()`.

Usage:

```
SharingChanges$new(
  profile,
  auth,
  identifier,
  starting_version = NULL,
  ending_version = NULL,
  starting_timestamp = NULL,
  ending_timestamp = NULL,
  columns = NULL,
  cache_path,
  concurrency
)
```

Arguments:

`profile`, `auth`, `identifier` Internal client state.

`starting_version`, `ending_version`, `starting_timestamp`, `ending_timestamp`, `columns`
Query options; see `SharingTable`'s `changes()` method.

`cache_path`, `concurrency` Internal table execution settings.

SharingClient

Delta Sharing client

Description

Delta Sharing client

Delta Sharing client

Details

A reusable client that owns a parsed profile and its authentication context. Discovery and table handles are created from the client. Download concurrency is configured on each table handle; query options live on its snapshot and changes readers.

Most users call `sharing_client()` rather than `SharingClient$new()`.

Methods

Public methods:

- `SharingClient$new()`
- `SharingClient$endpoint()`
- `SharingClient$list_shares()`
- `SharingClient$list_schemas()`
- `SharingClient$list_tables()`
- `SharingClient$table()`
- `SharingClient$print()`

Method `new()`: Create a client from a profile.

Usage:

`SharingClient$new(profile)`

Arguments:

`profile` Profile path or parsed list.

Method `endpoint()`: The configured profile endpoint.

Usage:

`SharingClient$endpoint()`

Returns: The endpoint URL string.

Method `list_shares()`: List available shares.

Usage:

`SharingClient$list_shares()`

Returns: A printable list of share records.

Method `list_schemas()`: List schemas in a share.

Usage:

`SharingClient$list_schemas(share)`

Arguments:

`share` Share name.

Returns: A printable list of schema records.

Method `list_tables()`: List tables in a share, optionally within one schema.

Usage:

`SharingClient$list_tables(share, schema = NULL)`

Arguments:

`share` Share name.

`schema` Optional schema name.

Returns: A printable list of table records.

Method `table()`: Create a reusable table handle.

Usage:

```
SharingClient$table(name, schema = NULL, share = NULL, concurrency = 4L)
```

Arguments:

name Table name, or a "share.schema.name" string when share and schema are omitted.
 schema Schema name when using explicit components.
 share Share name when using explicit components.
 concurrency Maximum number of data files downloaded concurrently.

Returns: A [SharingTable](#).

Method `print()`: Print the client.

Usage:

```
SharingClient$print(...)
```

Arguments:

... Ignored.

 SharingSnapshot

Delta Sharing snapshot reader

Description

Delta Sharing snapshot reader

Delta Sharing snapshot reader

Details

An immutable snapshot read specification with Arrow materializers. Created by `SharingTable$snapshot()`. Materialize with `to_arrow_stream()` (lazy), `to_arrow_reader()` (lazy), `to_arrow()`, `to_tibble()`, or `to_data_frame()`.

Super class

`delta.sharing::SharingReader` -> SharingSnapshot

Methods**Public methods:**

- [SharingSnapshot\\$new\(\)](#)

Method `new()`: Create a snapshot reader. Prefer `SharingTable$snapshot()`.

Usage:

```
SharingSnapshot$new(
  profile,
  auth,
  identifier,
  version = NULL,
  timestamp = NULL,
  columns = NULL,
  limit = NULL,
  predicate = NULL,
  response_format = "auto",
  cache_path,
  concurrency
)
```

Arguments:

profile, auth, identifier Internal client state.

version, timestamp, columns, limit, predicate, response_format Query options; see [SharingTable](#)'s snapshot() method.

cache_path, concurrency Internal table execution settings.

SharingTable	<i>Delta Sharing table handle</i>
--------------	-----------------------------------

Description

Delta Sharing table handle

Delta Sharing table handle

Details

A cheap, reusable reference to a shared table. Metadata methods query the control plane without scanning rows. snapshot() and changes() create reader objects that carry query options and expose materializers. Downloaded files are cached for the R session and shared by handles for the same table.

Created via SharingClient\$table(), not directly.

Predicate hints

Snapshot predicates are ordinary nested R lists. For example:

```
active_orders <- list(
  op = "equal",
  children = list(
    list(op = "column", name = "is_active", valueType = "bool"),
    list(op = "literal", value = "true", valueType = "bool")
  )
)
orders$snapshot(predicate = active_orders)
```

The sharing server treats the predicate as a best-effort hint, not an exact row filter.

Active bindings

`cache_path` Read-only path to this table's session cache. Removing it manually invalidates active lazy readers; the next read recreates it.

Methods

Public methods:

- `SharingTable$new()`
- `SharingTable$identifier()`
- `SharingTable$version()`
- `SharingTable$protocol()`
- `SharingTable$metadata()`
- `SharingTable$schema()`
- `SharingTable$snapshot()`
- `SharingTable$changes()`
- `SharingTable$print()`

Method `new()`: Create a table handle. Prefer `SharingClient$table()`.

Usage:

```
SharingTable$new(profile, auth, identifier, concurrency = 4L)
```

Arguments:

`profile` Parsed profile (internal).

`auth` Authentication context (internal).

`identifier` A list(share, schema, table) (internal).

`concurrency` Maximum number of concurrent file downloads.

Method `identifier()`: The structured table identifier.

Usage:

```
SharingTable$identifier()
```

Returns: A list with share, schema, and table.

Method `version()`: The current table version.

Usage:

```
SharingTable$version()
```

Returns: A non-negative whole number.

Method `protocol()`: The table protocol capabilities.

Usage:

```
SharingTable$protocol()
```

Returns: A safe list of protocol fields.

Method `metadata()`: The table metadata (no row scan).

Usage:

```
SharingTable$metadata()
```

Returns: A safe structured metadata list.

Method `schema()`: The table's logical schema.

Usage:

```
SharingTable$schema()
```

Returns: The parsed struct schema.

Method `snapshot()`: Configure a snapshot read.

Usage:

```
SharingTable$snapshot(  
  version = NULL,  
  timestamp = NULL,  
  columns = NULL,  
  limit = NULL,  
  predicate = NULL,  
  response_format = "auto"  
)
```

Arguments:

`version` Optional non-negative whole-number version.

`timestamp` Optional scalar timestamp string or POSIXct value.

`columns` Optional character vector of projected columns.

`limit` Optional non-negative whole-number row limit.

`predicate` Optional nested R list representing a structured server-side predicate hint. This is not an exact row filter.

`response_format` One of "auto", "delta", or "parquet".

Returns: A [SharingSnapshot](#).

Method `changes()`: Configure a change data feed read.

Usage:

```
SharingTable$changes(  
  starting_version = NULL,  
  ending_version = NULL,  
  starting_timestamp = NULL,  
  ending_timestamp = NULL,  
  columns = NULL  
)
```

Arguments:

`starting_version`, `ending_version` Optional version bounds.

`starting_timestamp`, `ending_timestamp` Optional timestamp strings or POSIXct bounds.

`columns` Optional character vector of projected columns.

Returns: A [SharingChanges](#).

Method `print()`: Print the table handle.

Usage:

```
SharingTable$print(...)
```

Arguments:

... Ignored.

sharing_client	Create a Delta Sharing client
----------------	-------------------------------

Description

Constructs a [SharingClient](#) from a Delta Sharing profile. The profile may be a path to a .share file or a parsed profile list. Construction parses and validates the profile but performs no network request or token exchange.

Usage

```
sharing_client(profile)
```

Arguments

profile	A profile file path or parsed profile list. Profile versions 1 (bearer) and 2 (bearer, basic, OAuth client-credentials, and private-key JWT) are supported.
---------	---

Value

A [SharingClient](#).

Examples

```
client <- sharing_client(demo_profile())
client$list_tables("delta_sharing")
```

Index

`delta.sharing::SharingReader`, [3](#), [5](#)
`demo_profile`, [2](#)

`sharing_client`, [9](#)
`sharing_client()`, [3](#)
`SharingChanges`, [2](#), [9](#)
`SharingClient`, [3](#), [9](#)
`SharingSnapshot`, [5](#), [8](#)
`SharingTable`, [3](#), [5](#), [6](#), [6](#)