Package 'lazysf'

November 14, 2020

Title Delayed Read for 'GDAL' Vector Data Sources

Version 0.1.0

Description Lazy read for drawings. A 'dplyr' back end for data sources supported by 'GDAL' vector drivers, that allows working with local or remote sources as if they are in-memory data frames. Basic features works with any drawing format ('GDAL vector data source') supported by the 'sf' package.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports sf (>= 0.7-0), methods, DBI, tibble, dbplyr, magrittr, dplyr

URL https://github.com/mdsumner/lazysf

BugReports https://github.com/mdsumner/lazysf/issues

Suggests knitr, rmarkdown

VignetteBuilder knitr

Collate 'SFSQLConnection.R' 'SFSQLDriver.R' 'SFSQLResult.R' 'connect.R' 'lazysf-package.R' 'lazysf.R' 'utils-pipe.R' 'zzz.R'

NeedsCompilation no

Author Michael Sumner [aut, cre] (<https://orcid.org/0000-0002-2471-7511>)

Maintainer Michael Sumner <mdsumner@gmail.com>

Repository CRAN

Date/Publication 2020-11-14 08:30:02 UTC

R topics documented:

`dbConnect,SFSQLDriver-method` *dbConnect*

Description

dbConnect for drawings that may be read by package sf

Usage

```
## S4 method for signature 'SFSQLDriver'
dbConnect(drv, DSN = "", readonly = TRUE, ...)
```

Arguments

`drv`	SFSQLDriver created by `SFSQL()`
`DSN`	data source name, may be a file, or folder path, database connection string, or URL
`readonly`	open in readonly mode (`TRUE` is the only option)
`...`	ignored

Details

The 'OGRSQL' available is documented with GDAL: https://gdal.org/user/ogr_sql_dialect.html

Examples

```
afile <- system.file("gpkg/nc.gpkg", package = "sf", mustWork = TRUE)
db <- dbConnect(SFSQL(), afile)
dbSendQuery(db, 'SELECT * FROM "nc.gpkg"')
```

`lazysf` *Delayed (lazy) read for GDAL vector*

Description

A lazy data frame for GDAL drawings ('vector data sources'). lazysf is DBI compatible and designed to work with dplyr. It should work with any data source (file, url, connection string) readable by the sf package function `sf_read`.

Usage

```
lazysf(x, layer, ...)

## S3 method for class 'character'
lazysf(x, layer, ..., query = NA)

## S3 method for class 'SFSQLConnection'
lazysf(x, layer, ..., query = NA)
```

Arguments

x	the data source name (file path, url, or database connection string • analogous to `sf::read_sf()` 'dsn')
layer	layer name (varies by driver, may be a file name without extension); in case `layer` is missing, `st_read` will read the first layer of `dsn`, give a warning and (unless `quiet = TRUE`) print a message when there are multiple layers, or give an error if there are no layers in `dsn`. If `dsn` is a database connection, then `layer` can be a table name or a database identifier (see `Id`). It is also possible to omit `layer` and rather use the `query` argument.
...	ignored
query	SQL query to pass in directly

Details

Lazy means that the usual behaviour of reading the entirety of a data source into memory is avoided. Printing the output results in a preview query being run and displayed (the top few rows of data).

The output of `lazysf()` is a 'tbl_SFSQLConnectionthat extendstbl_dbi' and may be used with functions and workflows in the normal DBI way, see `SFSQL()` for the lazysf DBI support.

The kind of q uery that may be run will depend on the type of format, see the list on the GDAL vector drivers page. For some details see the GDALSQL vignette.

When dplyr is attached the lazy data frame can be used with the usual verbs verbs (filter, select, distinct, mutate, transmute, arrange, left_join, pull, collect etc.). To see the result as a SQL query rather than a data frame preview use `dplyr::show_query()`.

To obtain an in memory data frame use an explict `collect()` or `st_as_sf()`. A call to `collect()` is triggered by `st_as_sf()` and will add the sf class to the output. A result may not contain a geometry column, and so cannot be convert to an sf data frame. Using `collect()` on its own returns an unclassed data.frame and may include a classed `sfc` geometry column.

As well as `collect()` it's also possible to use `tibble::as_tibble()` or `as.data.frame()` or `pull()` which all force computation and retrieve the result.

Value

a 'tbl_SFSQLConnection', extending 'tbl_lazy' (something that works with dplyr verbs, and only shows a preview until you commit the result via `collect()`) see Details

Examples

```
# online sources can work
geojson <- file.path("https://raw.githubusercontent.com/SymbolixAU",
                     "geojsonsf/master/inst/examples/geo_melbourne.geojson")

lazysf(geojson)


## normal file stuff
## (Geopackage is an actual database so with SELECT we must be explicit re geom-column)
f <- system.file("gpkg/nc.gpkg", package = "sf", mustWork = TRUE)
lazysf(f)
lazysf(f, query = "SELECT AREA, FIPS, geom FROM \"nc.gpkg\" WHERE AREA < 0.1")
lazysf(f, layer = "nc.gpkg") %>% dplyr::select(AREA, FIPS, geom) %>% dplyr::filter(AREA < 0.1)

## the famous ESRI Shapefile (not an actual database)
## so if we SELECT we must be ex
shp <- lazysf(system.file("shape/nc.shp", package = "sf", mustWork = TRUE))
library(dplyr)
shp %>%
 filter(NAME %LIKE% 'A%') %>%
 mutate(abc = 1.3) %>%
 select(abc, NAME, `_ogr_geometry_`) %>%
 arrange(desc(NAME))  #%>% show_query()

 ## a multi-layer file
 system.file("extdata/multi.gpkg", package = "lazysf", mustWork = TRUE)
```

SFSQL *SFSQL*

Description

SFSQL driver, use to dbConnect() to a data source readable by sf

Usage

```
SFSQL()
```

See Also

lazysf dbConnect

Examples

```
SFSQL()
```

st_as_sf *Force computation of a GDAL query*

Description

Convert lazysf to an in memory data frame or sf object

Usage

```
## S3 method for class 'tbl_SFSQLConnection'
st_as_sf(x, ...)

collect(x, ...)
```

Arguments

x	output of `lazysf()`
...	passed to `collect()`

Format

An object of class `function` of length 1.

Details

`collect()` retrieves data into a local table, preserving grouping and ordering.

`st_as_sf()` retrieves data into a local sf data frame (will succeed only if there is a geometry column of class `sfc`)

Value

a data frame from `collect()`, sf data frame from `st_as_sf()` (only if it contains an `sfc` geometry column)

See Also

lazysf

Examples

```
f <- system.file("gpkg/nc.gpkg", package = "sf", mustWork = TRUE)
lsf <- lazysf(f) %>% dplyr::select(AREA, FIPS, geom) %>% dplyr::filter(AREA < 0.1)
st_as_sf(lsf)
```

Index